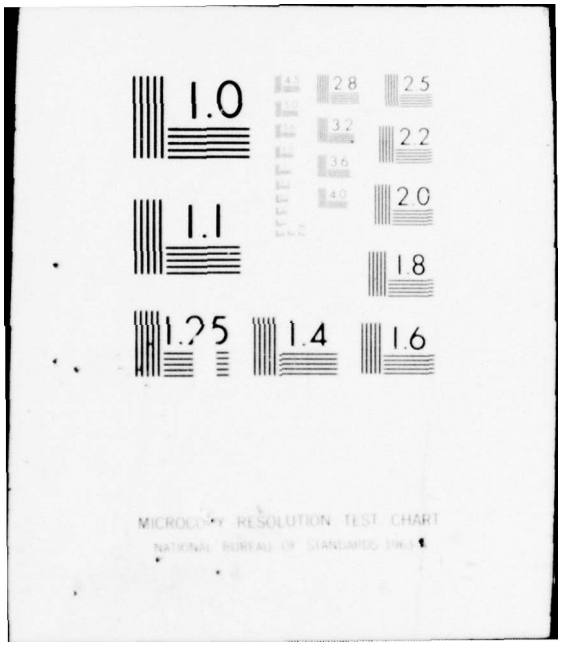
AD-A038 891 NAVAL SEA SYSTEMS COMMAND WASHINGTON D C F/G 17/1
NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK - NAVDAB - IN SUP--ETC(U)
DEC 75
UNCLASSIFIED NAVSEA-06H1/036-EVA/MOST- NL

1 OF 2
AD
A038891


1.0
1.1
1.25
1.4
1.6
2.8
2.5
3.2
2.2
3.6
2.0
4.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963

SEA 06H1/036-EVA/MOST-4

1 DECEMBER 1975

NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK

—NAVDAB—

IN SUPPORT OF

MOBILE SONAR TECHNOLOGY DEVELOPMENT

VOLUME 3

NAVAL SEA SYSTEMS COMMAND
DEPARTMENT OF THE NAVY
WASHINGTON, D.C. 20362

SEA 06H1/036-EVA/MOST-4

1 DECEMBER 1975

NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK

—NAVDAB—

IN SUPPORT OF

MOBILE SONAR TECHNOLOGY DEVELOPMENT

VOLUME 3

NAVAL SEA SYSTEMS COMMAND
DEPARTMENT OF THE NAVY
WASHINGTON, D.C. 20362

Approved for public release; distribution unlimited.

ADMINISTRATIVE STATEMENT

The NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB) is a joint effort of the Naval Undersea Center (NUC), the Naval Underwater Systems Center (NUSC), the Naval Research Laboratory (NRL), and the Naval Oceanographic Office (NAVOCEANO), under the sponsorship of the Sonar Technology Office of the Naval Sea Systems Command (Task Area SF 52 552 601). Each of these organizations is represented in the NAVDAB Steering Group. The Steering Group has the responsibility for development of the data bank, which has been installed at NUC/San Diego, California, NUSC/New London, Connecticut, and NAVOCEANO/ Washington, D.C.

The NAVDAB Steering Group is grateful to G. F. Anderson of Computer Sciences Corporation and G. E. Miller of Arthur D. Little, Inc., for their assistance on this project.

Released by:

L. A. Smothers (NUC)
NAVDAB Chairman

Sponsoring Authority:

C. D. Smith
NAVSEA

A. P. Franceschetti
NAVSEA

Steering Group

F. R. DiNapoli (NUSC)

K. V. Mackenzie (NAVOCEANO)

R. H. Ferris (NRL)

L. C. Maples (NUSC)

W. H. Geddes (NAVOCEANO)

S. R. Santaniello (NUSC)

R. F. Hosmer (NUC)

J. A. Whitney (NUC)

W. A. Kuperman (NRL)

(14)

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: NAVSEA-06H1/036-EVA/MOST-4	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER (9)
4. TITLE (and Subtitle): (6) NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK - NAVDAB - IN SUPPORT OF MOBILE SONAR TECHNOLOGY DEVELOPMENT, VOLUME 3.	5. TYPE OF REPORT & PERIOD COVERED: Final rept. / 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): NAVDAB Steering Group	8. CONTRACT OR GRANT NUMBER(s): SF 52-552-601
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Undersea Center, Code 3073, San Diego, CA. 92132	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: Task No. 19324
11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Sea Systems Command (NAVSEA 06H1-4), Washington, D.C. 20362	(11) 12. REPORT DATE: 1 Dec 1975 / 13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): (12) 129 p.	15. SECURITY CLASS. (of this report): Unclassified / 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report):

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report): (16) F52552 (17) SF52552601

18. SUPPLEMENTARY NOTES

Prepared in cooperation with the NAVDAB Steering Group.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Data bank
Underwater Acoustics
Environmental data
Computer Programs

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

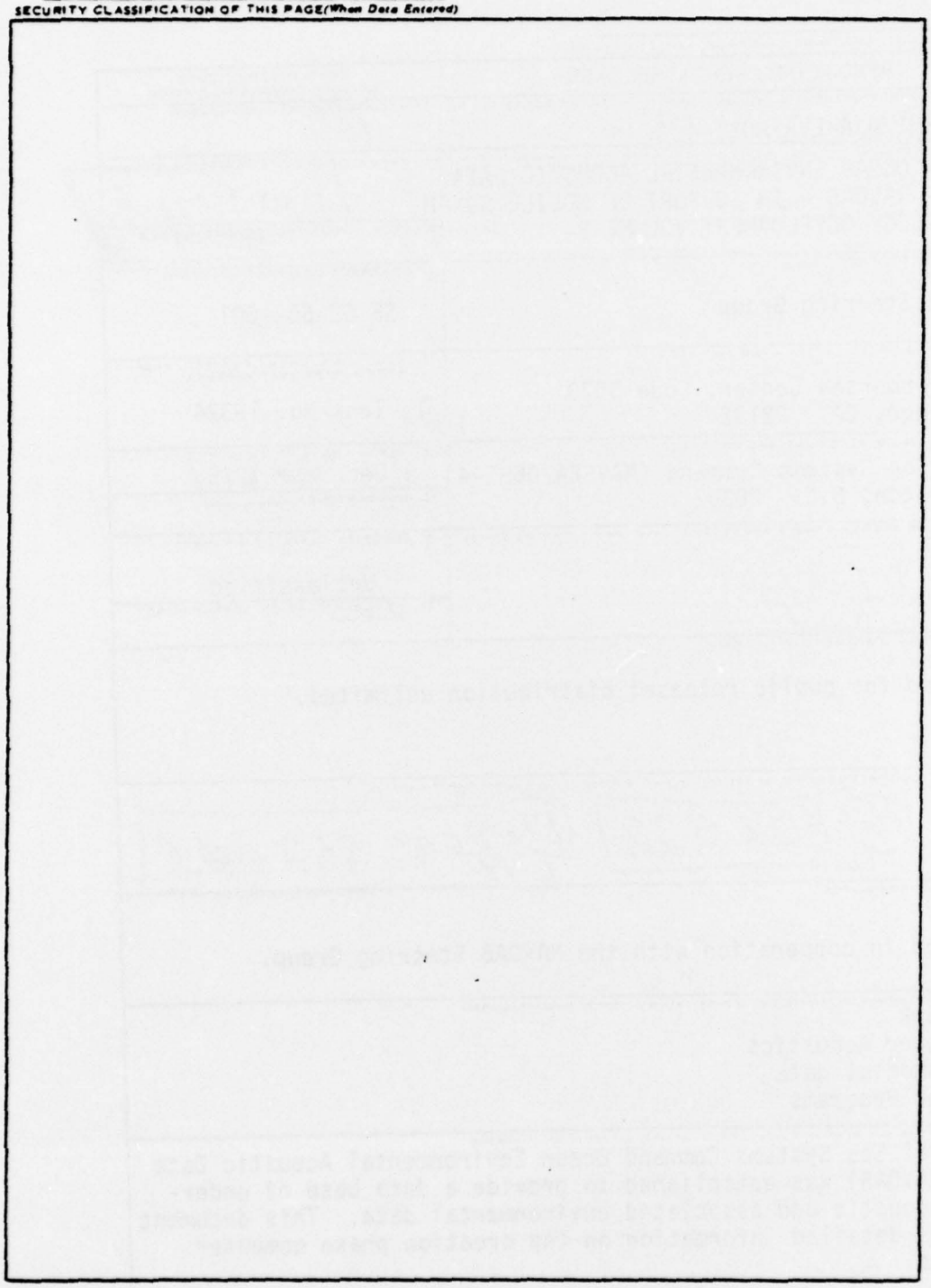
The Naval Sea Systems Command Ocean Environmental Acoustic Data Bank (NAVDAB) was established to provide a data base of underwater acoustic and associated environmental data. This document provides detailed information on the creation phase computer programs.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

391345

1/3



DEPARTMENT OF THE NAVY
NAVAL SEA SYSTEMS COMMAND
WASHINGTON, D.C. 20362

FOREWORD

The Naval Sea Systems Command Ocean Environmental Acoustic Data Bank (NAVDAB) was established to provide a data base for the development and validation of environmental acoustic models for mobile sonar application. NAVDAB is unique in that it is primarily for environmental and acoustic data taken concurrently and it is the only data bank of this type operated for mobile sonar application.

A set of five reports have been prepared to describe how NAVDAB works and how to use it. This is volume 3 of the set. The individual reports cover the following topics:

Volume 1 User's Guide to Retrieval
Volume 2 Input Format Guide
Volume 3 Details of Creation Phase
Volume 4 Details of Retrieval Phase
Volume 5 Details of Miscellaneous Support Programs

Governmental and industrial activities desiring to submit or retrieve environmental acoustic data should contact:

Naval Undersea Center
Code 3073
San Diego, CA - 92132

Naval Underwater Systems Center
Code TA1A
New London Laboratory
New London, CT - 06320

Naval Oceanographic Office
Code 3440
Washington, D.C. - 20373

Copies of the five reports can be obtained through the Defense Documentation Center, Defense Supply Agency, Cameron Station, Alexandria, VA - 22314.

C.D. Smith, Director
Sonar Technology Office,
06H1/036

CONTENTS

LIST OF ILLUSTRATIONS

SECTION 1.0

INTRODUCTION

The Naval Sea Systems Command (NAVSEA) Ocean Environmental Acoustic Data Bank (NAVDAB) was established to provide a data base for the development and validation of environmental acoustic models for mobile sonar applications. The purpose of this document is to provide details of the computer programs which generate the NAVDAB data base.

This is Volume 3 of the five-volume set covering the NAVDAB computer program documentation. The individual volumes cover the following topics:

Volume 1. User's Guide to Retrieval

Volume 2. Input Format Guide

Volume 3. Details of Creation Phase

Volume 4. Details of Retrieval Phase

Volume 5. Details of Miscellaneous Support Programs

Copies of these documents may be obtained from the Defense Documentation Center.

SECTION 2.0

DESCRIPTION OF THE DATA BASE

Many factors affect the design of a data base and retrieval system. In the case of NAVDAB a large number and variety of data parameters were considered. To meet this requirement an open-ended tabular format, with variable record length, was selected to accommodate the initial sets of parameters and data and allow for expansion.

Another primary consideration in the data base design was the need for rapid access to all elements of all data. The data base must be suited to a versatile, efficient, random-access, retrieval system. Random-access retrieval techniques dictate, to a great extent, how data are structured on a storage device.

Other requirements of the NAVDAB system include: safeguard of classified information; allowance for operation in either batch or time-sharing mode; ease of updating and maintenance; and use of a universally-accepted computer language, as well as sufficient modularity, for ready adaptation to the different types of computers on which the system may be implemented.

In view of all these considerations, the data organization shown in Figure 2-1 was developed. It is a multilevel, hierarchically-organized, partially circular, linked-list structure, with large-capacity disk for primary storage. Four levels are involved:

1. EXPERIMENT — The highest level, pertains to the overall program of measurements — such as AMOS, FASOR I (Forward Area Sonar Research)

2. CRUISE — The second level is applicable to natural subsets of an EXPERIMENT, such as individual AMOS cruises or measurements taken within a specified area

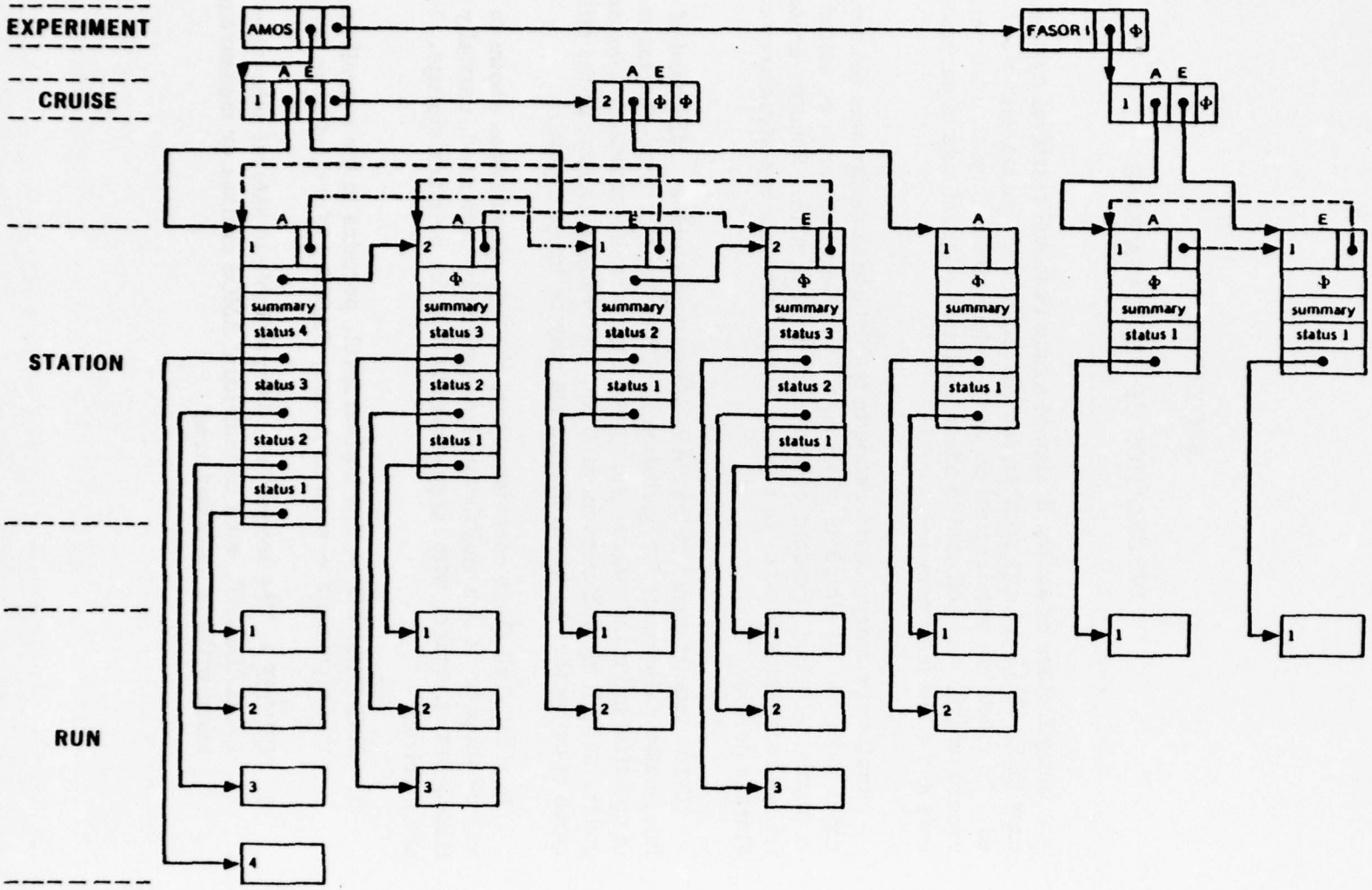


Figure 2-1. NAVDAB Data Base Structure.

3. STATION – A subdivision of a CRUISE: refers to measurements at or about a specific geographical location, such as FASOR II, Cruise 1, Station 1

4. RUN – The fourth and lowest level: designates a discrete set of measurements forming an element of a STATION, such as a propagation loss run or a hydrographic cast

Actual numerical data are stored only at the RUN level, which contains no other information. Key catalog information about the various Runs of a Station are stored at the STATION level, which contains, for all the Runs, chronological and geographical data and access category, and lists the acoustic and environmental parameters. The organization of the data is described for each Run, including the Run's storage location on the disk. Locations of associated Stations are also included. There are two types of Stations, acoustic and environmental, but the only environmental data currently approved by the Steering Group are those supporting accepted acoustic data.

Catalog information for the Stations of a Cruise is stored at the CRUISE level, which also contains storage locations of the first acoustic and environmental Stations and the next Cruise of the Experiment. In addition, there is an alphanumeric description of the features of the particular Cruise. These notes may be printed out in the retrieval process as background information.

The EXPERIMENT level contains summary information for the Experiment and directs flow down through the other levels. This flexible pointer structure allows additions or deletions of data without changing the programs that access the data.

SECTION 3.0

DESCRIPTION OF THE DATA ENTRY PROCESS

The program INIT0 is used to initialize the NAVDAB data base. The actual generation of the data base is accomplished by program MAIN. After examining the data base to determine the current status of its contents, MAIN begins reading card images which were prepared according to established NAVDAB input formats. (See Volume 2 of this documentation series.)

As each new Experiment, Cruise, or Station is read in, links (pointers) are established to associate it on the data base with the previous Experiment, Cruise, or Station. The schemes for processing constants, parameters, and variables are the same for environmental input card images as for acoustic card images. Numerous error checks are made as the data are processed.

Flowcharts for the data entry process are shown in Figure 3-1 through 3-3. Complete computer program listings are contained in Appendix A.

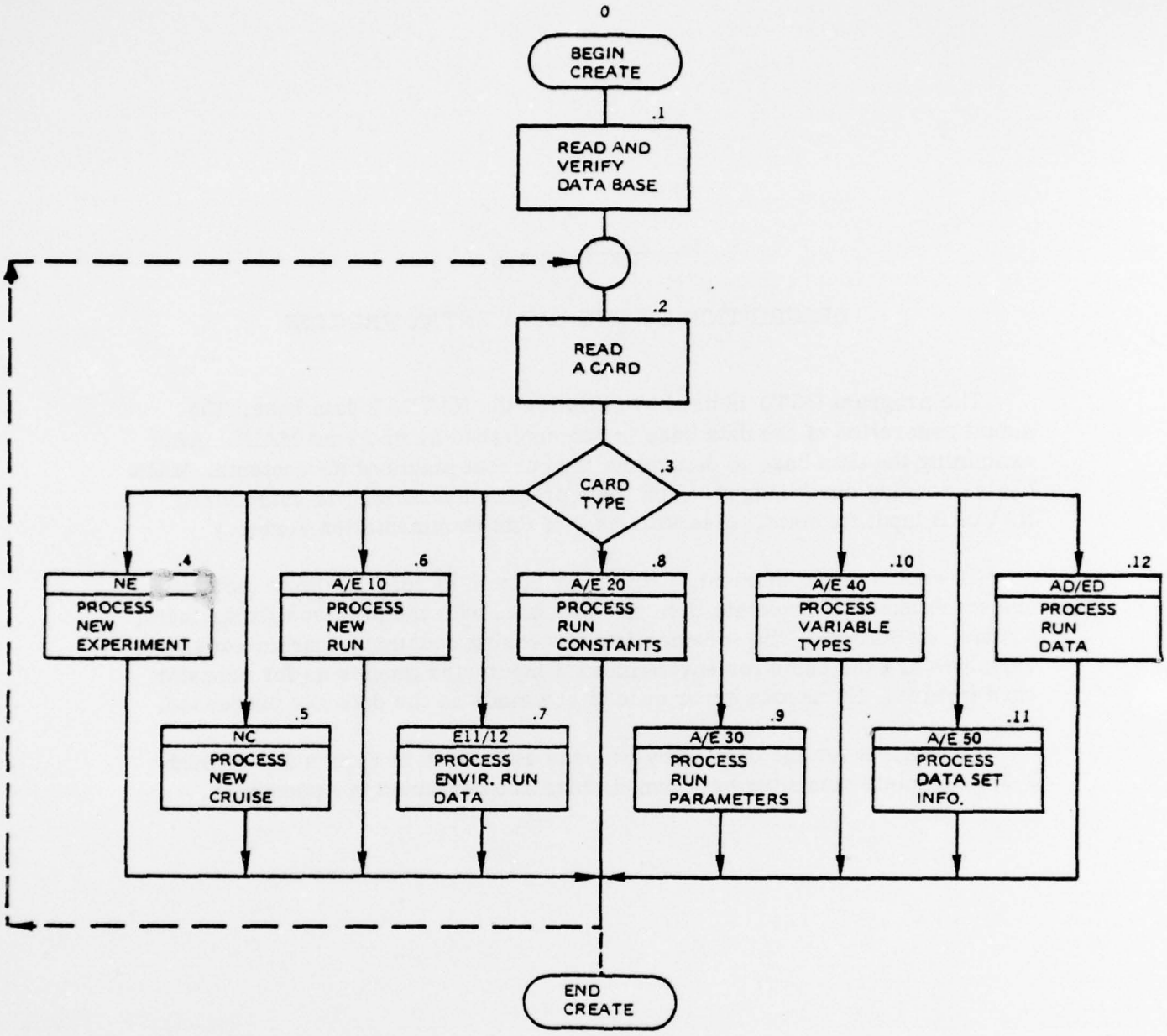

0
BEGIN CREATE
.1
READ AND VERIFY DATA BASE
.2
READ A CARD
.3
CARD TYPE
.4
NE
PROCESS NEW EXPERIMENT
.5
NC
PROCESS NEW CRUISE
.6
A/E 10
PROCESS NEW RUN
.7
E11/12
PROCESS ENVIR. RUN DATA
.8
A/E 20
PROCESS RUN CONSTANTS
.9
A/E 30
PROCESS RUN PARAMETERS
.10
A/E 40
PROCESS VARIABLE TYPES
.11
A/E 50
PROCESS DATA SET INFO.
.12
AD/ED
PROCESS RUN DATA
END CREATE

Figure 3-1.

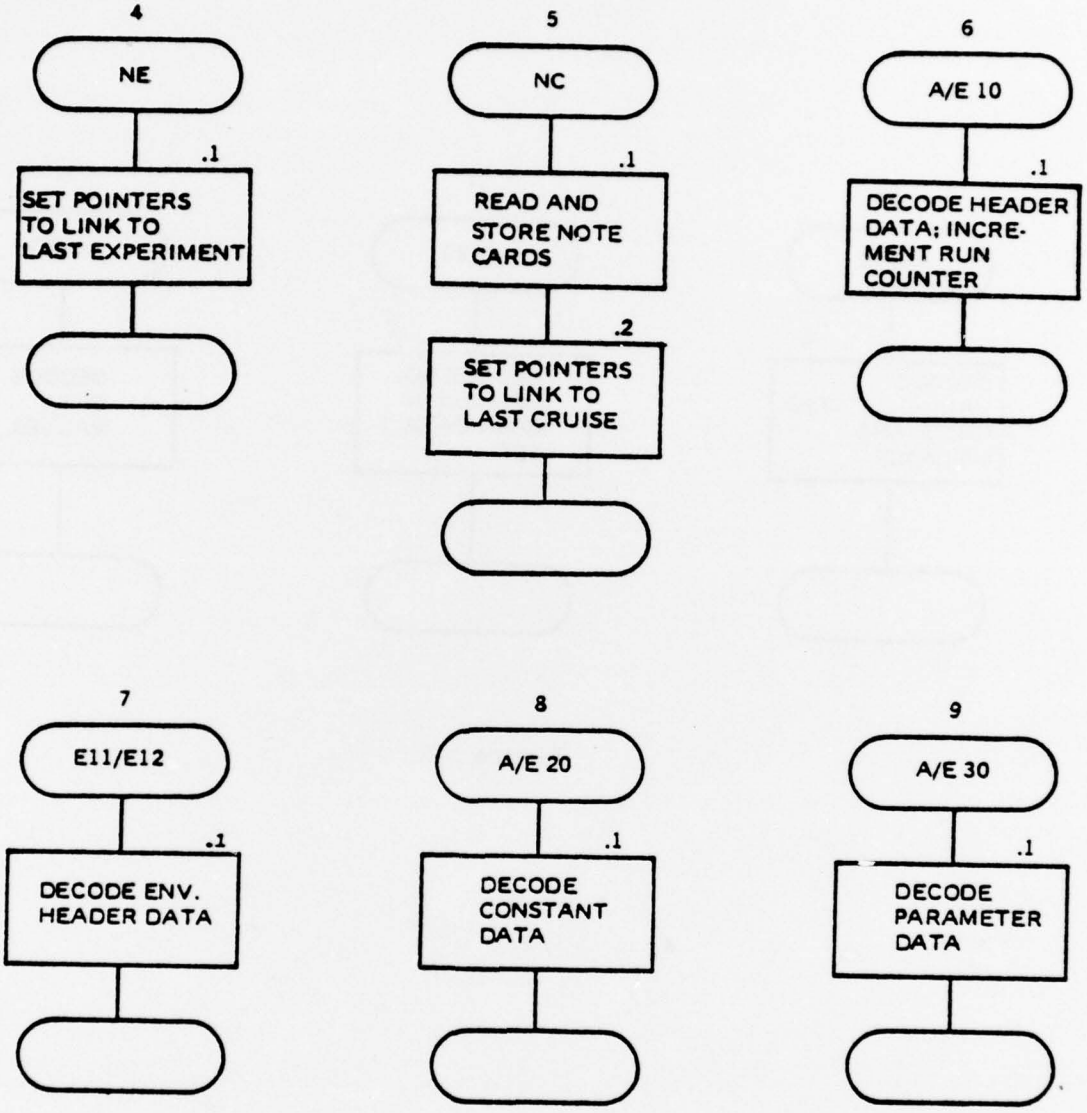

4
NE
.1
SET POINTERS TO LINK TO LAST EXPERIMENT
5
NC
.1
READ AND STORE NOTE CARDS
.2
SET POINTERS TO LINK TO LAST CRUISE
6
A/E 10
.1
DECODE HEADER DATA; INCRE-MENT RUN COUNTER
7
E11/E12
.1
DECODE ENV. HEADER DATA
8
A/E 20
.1
DECODE CONSTANT DATA
9
A/E 30
.1
DECODE PARAMETER DATA

Figure 3-2.

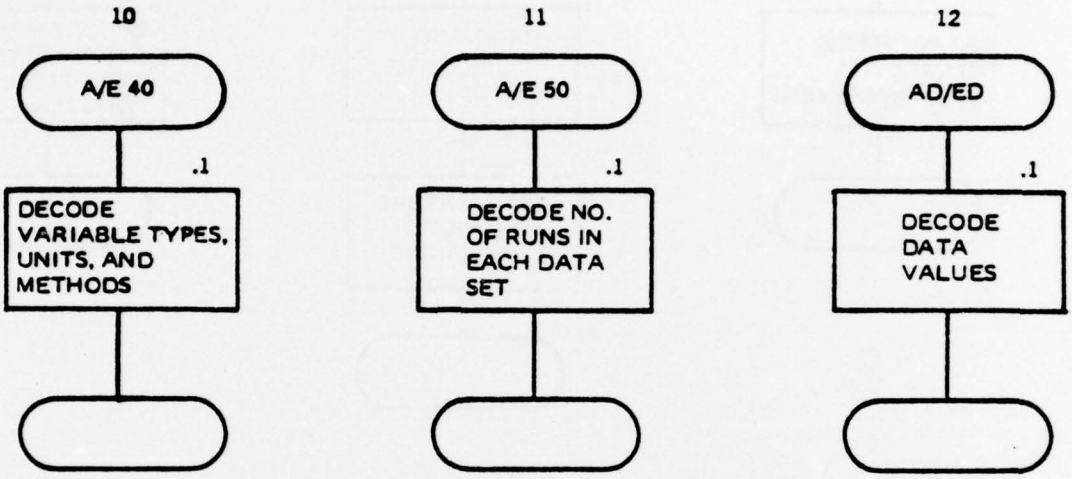

10
A/E 40
.1
DECODE VARIABLE TYPES, UNITS, AND METHODS
11
A/E 50
.1
DECODE NO. OF RUNS IN EACH DATA SET
12
AD/ED
.1
DECODE DATA VALUES

Figure 3-3.

APPENDIX A

COMPUTER PROGRAM LISTINGS

ELEMENT TABLE

NAME	PAGE	TYPE	DATE	TIME	SEQ #	SIZE-PRE, TEXT (CYCLE WORD) PSHMODE	LOCATION
ACMPRS	A-5	FOR SYMB	16 DEC 75	10:13:24	1	34 S 0 1	1792
ACMPRS		RELOCATABLE	16 DEC 75	16:09:24	2	2 20	1826
ACPRES	A-9	FOR SYMB	22 AUG 74	10:23:46	3	12 S 0 1	1848
ACPRES		RELOCATABLE	16 DEC 75	16:09:37	4	1 8	1860
ACSTAT	A-11	FOR SYMB	16 DEC 75	10:14:16	5	15 S 0 1	1869
ACSTAT		RELOCATABLE	16 DEC 75	16:09:49	6	1 10	1884
ANGLES	A-13	FOR SYMB	01 NOV 74	12:02:12	7	13 S 0 1	1895
ANGLES		RELOCATABLE	16 DEC 75	16:09:52	8	1 10	1908
ASCMPS	A-15	FOR SYMB	16 DEC 75	10:14:48	9	7 S 0 1	1919
ASCMPS		RELOCATABLE	16 DEC 75	16:10:04	10	1 5	1926
CONVRT	A-16	FOR SYMB	31 OCT 75	15:40:50	11	27 S 0 1	1932
CONVRT		RELOCATABLE	16 DEC 75	16:10:34	12	3 25	1959
COPAVA	A-19	FOR SYMB	16 DEC 75	10:19:39	13	55 S 0 1	1987
COPAVA		RELOCATABLE	16 DEC 75	16:10:25	14	2 44	2042
CRDRED	A-25	FOR SYMB	28 MAY 75	20:43:35	15	4 S 0 1	2088
CRDRED		RELOCATABLE	16 DEC 75	16:10:37	16	1 4	2092
CRUINT	A-26	FOR SYMB	11 JUL 74	09:41:50	17	7 S 0 1	2097
CRUINT		RELOCATABLE	16 DEC 75	16:10:37	18	1 2	2104
CRUSUM	A-27	FOR SYMB	16 DEC 75	10:24:41	19	20 S 0 1	2107
CRUSUM		RELOCATABLE	16 DEC 75	16:10:51	20	2 12	2127
CYCLES	A-29	FOR SYMB	24 JUN 74	14:33:34	21	12 S 0 1	2141
CYCLES		RELOCATABLE	16 DEC 75	16:10:51	22	1 8	2153
DBLOSS	A-31	FOR SYMB	06 SEP 74	09:20:35	23	15 S 0 1	2162
DBLOSS		RELOCATABLE	16 DEC 75	16:11:03	24	1 9	2177
DEEBEE	A-33	FOR SYMB	30 OCT 75	08:46:09	25	19 S 0 1	2187
DEEBEE		RELOCATABLE	16 DEC 75	16:11:01	26	1 15	2206
DENSTY	A-35	FOR SYMB	28 JUN 74	13:34:17	27	12 S 0 1	2222
DENSTY		RELOCATABLE	16 DEC 75	16:11:13	28	1 7	2234
DSKRED	A-37	FOR SYMB	16 DEC 75	10:24:49	29	3 S 0 1	2242
DSKRED		RELOCATABLE	16 DEC 75	16:11:24	30	1 4	2245
DSKRIT	A-38	FOR SYMB	16 DEC 75	10:25:04	31	3 S 0 1	2250
DSKRIT		RELOCATABLE	16 DEC 75	16:11:28	32	1 4	2253
EAST	A-39	ELT SYMB	16 DEC 75	16:09:03	33	2 S 0 1	2258
EAST		RELOCATABLE	16 DEC 75	15:56:52	34	1 3	2260
ENERGY	A-40	FOR SYMB	24 JUN 74	14:33:40	35	13 S 0 1	2264
ENERGY		RELOCATABLE	16 DEC 75	16:11:27	36	1 8	2277
ENSTAT	A-42	FOR SYMB	16 DEC 75	10:25:04	37	15 S 0 1	2286
ENSTAT		RELOCATABLE	16 DEC 75	16:11:38	38	1 10	2301
EXPINT	A-44	FOR SYMB	16 DEC 75	10:25:15	39	5 S 0 1	2312
EXPINT		RELOCATABLE	16 DEC 75	16:11:37	40	1 2	2317
EXPSUM	A-45	ELT SYMB	16 DEC 75	16:09:15	41	18 S 0 1	2320
EXPSUM		RELOCATABLE	16 DEC 75	15:56:48	42	2 11	2338
INITD	A-47	FOR SYMB	16 DEC 75	10:25:52	43	6 S 0 1	2351
INITD		RELOCATABLE	16 DEC 75	16:11:53	44	1 6	2357
INTENS	A-48	FOR SYMB	24 JUN 74	14:34:29	45	12 S 0 1	2364
INTENS		RELOCATABLE	16 DEC 75	16:12:00	46	1 7	2376
LINEAR	A-50	FOR SYMB	30 AUG 74	13:34:51	47	15 S 0 1	2384
LINEAR		RELOCATABLE	16 DEC 75	16:12:04	48	1 11	2399
LNGTME	A-52	FOR SYMB	24 JUN 74	14:32:36	49	12 S 0 1	2411
LNGTME		RELOCATABLE	16 DEC 75	16:12:03	50	1 8	2423
MAIN	A-54	FOR SYMB	16 DEC 75	10:26:28	51	276 S 0 1	2432
MAIN		RELOCATABLE	16 DEC 75	16:12:26	52	4 158	2708
MAXMIN	A-77	FOR SYMB	16 DEC 75	10:26:48	53	11 S 0 1	2870
MAXMIN		RELOCATABLE	16 DEC 75	16:12:25	54	6	2881

	PAGE										
MOVINT	A-78	ELT SYMB	16 DEC 75	16:09:13	55		11	5	0	1	2888
MOVINT		RELOCATABLE	16 DEC 75	15:52:49	56	1	6				2899
NSCTRS	A-79	FOR SYMB	16 DEC 75	15:26:28	57		4	5	0	1	2906
NSCTRS		RELOCATABLE	16 DEC 75	16:12:37	58	1	2				2910
PCKCRU	A-80	FOR SYMB	16 DEC 75	10:27:12	59		13	5	0	1	2913
PCKCRU		RELOCATABLE	16 DEC 75	16:12:36	60	1	8				2926
PCKEXP	A-82	FOR SYMB	16 DEC 75	10:27:14	61		9	5	0	1	2935
PCKEXP		RELOCATABLE	16 DEC 75	16:12:49	62	1	6				2944
PERCNT	A-83	FOR SYMB	17 JUL 75	13:52:12	63		12	5	0	1	2951
PERCNT		RELOCATABLE	16 DEC 75	16:13:02	64	1	8				2963
PKAREA	A-85	FOR SYMB	16 DEC 75	16:00:29	65		13	5	0	1	2972
PKAREA		RELOCATABLE	16 DEC 75	16:13:04	66	1	8				2985
PLANAR	A-87	FOR SYMB	03 SEP 75	09:15:53	67		13	5	0	1	2994
PLANAR		RELOCATABLE	16 DEC 75	16:13:14	68	1	9				3007
POWOWR	A-89	FOR SYMB	24 JUN 74	14:34:03	69		12	5	0	1	3017
POWOWR		RELOCATABLE	16 DEC 75	16:13:13	70	1	7				3029
PRESSR	A-91	FOR SYMB	22 AUG 74	10:25:08	71		14	5	0	1	3037
PRESSR		RELOCATABLE	16 DEC 75	16:13:26	72	1	10				3051
PRINTO	A-93	FOR SYMB	16 DEC 75	10:27:25	73		10	5	0	1	3062
PRINTO		RELOCATABLE	16 DEC 75	16:13:37	74	2	11				3072
PRSLVL	A-94	FOR SYMB	22 AUG 74	11:09:28	75		12	5	0	1	3085
PRSLVL		RELOCATABLE	16 DEC 75	16:13:39	76	1	7				3097
SHRTME	A-96	FOR SYMB	24 JUN 74	14:32:29	77		12	5	0	1	3105
SHRTME		RELOCATABLE	16 DEC 75	16:13:51	78	1	9				3117
SPEEDS	A-98	FOR SYMB	26 JUN 74	21:22:11	79		13	5	0	1	3127
SPEEDS		RELOCATABLE	16 DEC 75	16:13:49	80	1	10				3140
STOR	A-100	FOR SYMB	19 FEB 75	09:22:54	81		6	5	0	1	3151
STOR		RELOCATABLE	16 DEC 75	16:14:01	82	1	2				3157
TMPTUR	A-101	FOR SYMB	17 JUL 75	13:45:33	83		13	5	0	1	3160
TMPTUR		RELOCATABLE	16 DEC 75	16:14:02	84	1	11				3173
UNPKCR	A-103	ELT SYMB	16 DEC 75	16:09:14	85		10	5	0	1	3185
UNPKCR		RELOCATABLE	16 DEC 75	15:54:28	86	1	6				3195
UNPKXP	A-104	ELT SYMB	16 DEC 75	16:09:15	87		7	5	0	1	3202
UNPKXP		RELOCATABLE	16 DEC 75	15:54:27	88	1	5				3209
VOLUME	A-105	FOR SYMB	24 JUN 74	14:32:21	89		13	5	0	1	3215
VOLUME		RELOCATABLE	16 DEC 75	16:14:15	90	1	8				3228
%EST	A-107	ELT SYMB	16 DEC 75	16:09:16	91		2	5	0	1	3237
%EST		RELOCATABLE	16 DEC 75	15:55:02	92	1	3				3239
											3243

DATA ENTRY PROGRAM ELEMENT TABLE AND INDEX TO LISTINGS (CONTINUED)

CALLED ROUTINES

CALLING ROUTINES

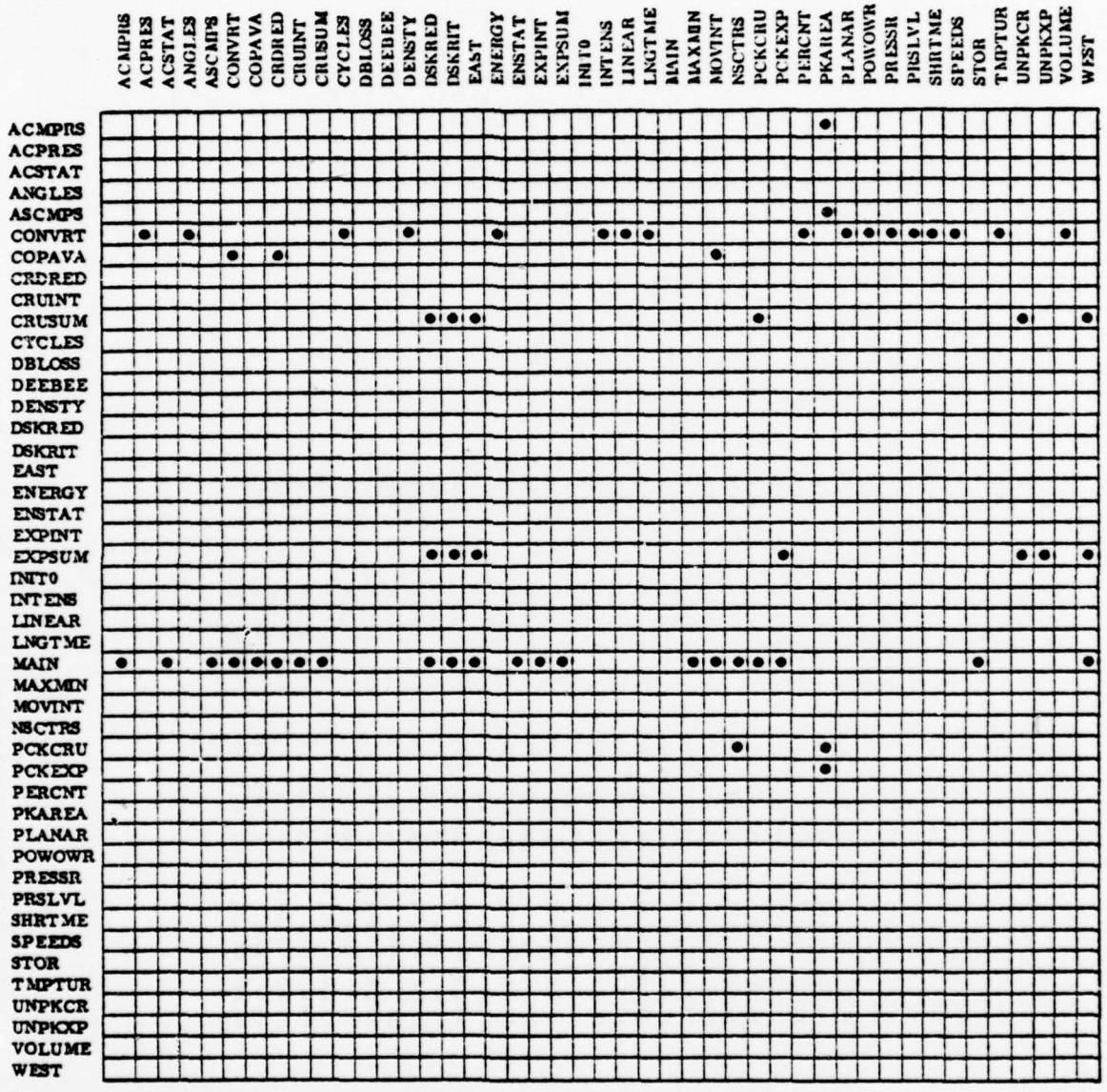


Figure A-1. Data Entry Program Cross-Reference Map.

```
@ELT,L     N,ACMPRS,,ACMPRS

      SUBROUTINE ACMPRS (BFR,IPTR,LNTH)
C
C
      IMPLICIT INTEGER (A-Z)
C
C
C
      COMMON /RUN   /PORA,NILAT,NINS,NFLAT,NFNS,NILON,NIEW,NFLON,NFEW,
     *               NIMO,NIDAY,NIYR,NITIM,NIZON,NFMO,NFDAY,NFYR,NFTIM,
     *               NFZON,NAVCOD,DTASRC,SYSSRC,SYSNCH,SRCTYP,RCRTYP,
     *               CLAS,DESCR,NAVDIR,NAVHT,NIUNIT,NAVPRD,PRDUNT,
     *               SEASTA,NNDDIR,NNDVEL,VELUNT,SWLDIR,SWLHT,
     *               SHTUNT,SWLPRD,SPDUNT,WEATHR,BOTDPH,DPHUNT,BTMSLP,
     *               BATHY,INFO(9),RSTAT(2),RUNNO
      REAL NAVDIR,NAVHT,NAVPRD,SEASTA,NNDDIR,NNDVEL,SWLDIR,
     *     SWLHT,SWLPRD,WEATHR,BOTDPH,BTMSLP
C
C
C
C
      COMMON /CPV   /NCON,CON(9,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
     *               NDATS,NRUNS(50),NDATA,VALRNG(2,30,50),DATA(30000)
      DIMENSION ICON(9,30),IPAR(30,30),IVLRNG(2,30,50)
      EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
     *            (VALRNG(1,1,1),IVLRNG(1,1,1))
      REAL CON,PAR,DATA,VALRNG
C
C
      DIMENSION BFR(1)
      REAL BFR
C
C           SET ISUB AS WORKING SUBSCRIPT. IPTR NOT CHANGED UNTIL RETURN
C
      ISUB = IPTR
C
C
C
      FLD(1),18,BFR(ISUB))=RUNNO
C
C
      ISUB=ISUB + 1
      FLD ( 0,9,BFR(ISUB)) = NAVCOD
      FLD ( 9,9,BFR(ISUB)) = DTASRC
      FLD (18,9,BFR(ISUB)) = CLAS
      FLD (27,9,BFR(ISUB)) = DESCR
C
      ISUB = ISUB + 1
C
      FLD ( 0,9,BFR(ISUB)) = SYSSRC
      FLD ( 9,9,BFR(ISUB)) = SYSNCH
      FLD (18,9,BFR(ISUB)) = SRCTYP
      FLD (27,9,BFR(ISUB)) = RCRTYP
C
C           COMPRESS AREA/TIME OF RUN
C
```

```
      ISUB = ISUB +1
      CALL PKAREA (PORA,RILAT,RILON,RIDAY,RITIM,RIZON,
     .                  RFLAT,RFLON,RFDAY,RFTIM,RFZON,BFR(ISUB))
C
C           MOVE STATUS BITS
C
      ISUB = ISUB + 4
      FLD(0,36,BFR(ISUB  )) = RSTAT(1)
      FLD(0,36,BFR(ISUB+1)) = RSTAT(2)
C
C           INIT PTR TO RUN DATA
C
      ISUB = ISUB +2
      BFR(ISUB) = 0
C
C           THIS ENDS THE PACKING OF DATA FROM /RUN/. FROM HERE ON
C           THE DATA IS PACKED FROM COMMON /CPV/.
C
C
      ISUB = ISUB +1
      FLD( 3,9,BFR(ISUB)) = NCON
      FLD( 9,9,BFR(ISUB)) = NPAR
      FLD(18,9,BFR(ISUB)) = NVAR
      FLD(27,9,BFR(ISUB)) = NDATS
C
C                 PACK NAME CODE & PROCESS CODE FOR ALL CONSTANTS
C
      IF (NCON .EQ. 0) GO TO 45
      DO 30 I=1,NCON,2
      ISUB = ISUB +1
      BFR(ISUB) = 0
      FLD( 0,8,BFR(ISUB)) = ICON(1,I)
      FLD( 8,10,BFR(ISUB)) = ICON(4,I)
      J = I+1
      IF (J .GT. NCON) GO TO 30
      FLD(18,8,BFR(ISUB)) = ICON(1,J)
      FLD(26,10,BFR(ISUB)) = ICON(4,J)
   30 CONTINUE
C
C
      DO 40 I=1,NCON
      ISUB = ISUB +1
      BFR(ISUB) = CON(3,I)
   40 CONTINUE
C
C                 PACK DATA FOR ALL PARAMETERS
C
C
   45 IF (NPAR .EQ. 0) GO TO 65
      DO 60 I=1,NPAR
C
C                   PACK DATA FOR PARAMETER(I)
C
      ISUB = ISUB +1
      FLD( 0,9,BFR(ISUB)) = IPAR(1,I)
      FLD( 9,8,BFR(ISUB)) = IPAR(2,I)
      FLD(17,10,BFR(ISUB)) = IPAR(4,I)
C
```

```
C                    PACK ALL VALUES FOR PARAMETER(I)
C
      JJ = IPAR(1,I)
           DO 50 J=1,JJ
           ISUB = ISUB +1
           BFR(ISUB) = PAR(1+J,I)
   50      CONTINUE
C
   60 CONTINUE
C
C                    PACK NAME & PROCESS CODES FOR ALL VARIABLES
C
   65 IF (NVAR .EQ. 0) GO TO 101
      DO 75 I=1,NVAR,2
      ISUB = ISUB +1
      BFR(ISUB) = 0
      FLD( 0,6,BFR(ISUB)) = IVAR(1,I)
      FLD( 8,10,BFR(ISUB)) = IVAR(3,I)
      J = I + 1
      IF (J .GT. NVAR) GO TO 70
      FLD(18,6,BFR(ISUB)) = IVAR(1,J)
      FLD(26,10,BFR(ISUB)) = IVAR(3,J)
   70 CONTINUE
C
C            PACK NROWS IN EACH DATA SET INTO BFR.
C
      DO 75 I=1,NDATS,2
      ISUB = ISUB+1
      BFR(ISUB) = 0
      FLD( 0,18,BFR(ISUB)) = NROWS(I)
      J = I+1
      IF (J .GT. NDATS) GO TO 75
      FLD(18,19,BFR(ISUB)) = NROWS(J)
   75 CONTINUE
C
C
      DO 100 K=1,NDATS
C
           DO 90 J=1,NVAR
C
                DO 80 I=1,2
                ISUB = ISUB +1
                BFR(ISUB) = VALRNG(I,J,K)
   80           CONTINUE
C
   90      CONTINUE
C
  100 CONTINUE
C
C          COMPUTE 'LNTH'
C
  101 LNTH = ISUB+1 - IPTR
C
C
      FLD(18,14,BFR(IPTR)) = ISUB +1
      IPTR = ISUB +1
C
```

```
169.      C          ALL DONE !
170.      C
171.                 RETURN
172.                 END
```

```
QFOR,SW      N.ACPRES,.ACPRES

      SUBROUTINE ACPRES(VALUE,UNICDE,UNITS)
C
C     TABLE OF ACPRES MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,3)/48,49,50/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,3)/'MICRO ','BARS  ',
     .                'PASCAL','S     ','MICRO ','PAS   '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='MICRO '
      UNITS(4)='PASCAL'
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-50
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,3)/ 1.D5,1.D3,1D0/
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,3
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
    2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
```

```
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN ACOUSTIC PRESSURE TABLE,')
      RETURN
      END
```

```
      SUBROUTINE ACSTAT (STATUS)
C
C
C
      IMPLICIT INTEGER (A-Z)
C
C
      COMMON /CPV   /NCON,CON(4,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
     .              NDATS,NROWS(50),NDATA,VALRNG(2,30,50),DATA(30000)
      DIMENSION ICON(4,30),IPAR(30,30),IVLRNG(2,30,50)
      EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
     .            (VALRNG(1,1,1),IVLRNG(1,1,1))
      REAL CON,PAR,DATA,VALRNG
C
C
      DIMENSION STATUS(2)
C
C         CLEAR STATUS
C
      STATUS(1) =0
      STATUS(2) =0
C
C         CHECK CONSTANTS
C
      IF (NCON .EQ. 0) GO TO 50
C
C         PROCESS CONSTANTS
C
      DO 40 I=1,NCON
C
      ICOD = IABS(ICON(1,I) -1)
C
      ISUB = 1
      IF (ICOD .LE. 35) GO TO 10
      ISUB = 2
      ICOD = IABS(ICOD -36)
C
      IF (ICOD .LT. 35) GO TO 10
      WRITE (6,4) ICON(1,I)
    4 FORMAT (' ERROR -- ACSTAT - ILLEGAL CODE',I10)
      STOP
C
   10 FLD(ICOD,1,STATUS(ISUB)) = 1
   40 CONTINUE
C
C         CHECK PARAMETERS
C
C
   50 IF (NPAR .EQ. 0) GO TO 70
C
C         PROCESS PARAMETERS
C
C
      DO 65 I=1,NPAR
C
      ICOD = IABS(IPAR(2,I) -1)
```

```
      C
            ISUB = 1
            IF (ICOD .LE. 35) GO TO 60
            ISUB = 2
            ICOD = IABS(ICOD -36)
      C
            IF (ICOD .LT. 35) GO TO 60
            WRITE (6,4) IPAR(2,I)
            STOP
      C
         60 FLD(ICOD,1,STATUS(ISUB)) = 1
         65 CONTINUE
      C
      C           CHECK VARIABLES
      C
         70 IF (NVAR .EQ. 0) GO TO 100
      C
      C           PROCESS VARIABLES
      C
            DO 90 I=1,NVAR
      C
            ICOD = IABS(IVAR(1,I) -1)
      C
            ISUB = 1
            IF (ICOD .LE. 35) GO TO 80
            ISUB = 2
            ICOD = IABS(ICOD -36)
      C
            IF (ICOD .LT. 35) GO TO 80
            WRITE (6,4) IVAR(1,I)
            STOP
      C
         80 FLD(ICOD,1,STATUS(ISUB)) = 1
         90 CONTINUE
      C
        100 RETURN
            END
```

```
@FOR,SN     N.ANGLES,.ANGLES

      SUBROUTINE ANGLES(VALUE,UNICDE,UNITS)
C
C     TABLE OF ANGLES MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,3)/37,38,85/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,3)/'RADIAN','S     ',
     *                'DEGREE','S     ','FTHMS/','NTCLML'/
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,2)/1.0D0,.0174532917D0/
C
C     ENTER INTERNAL DEGREE UNITS.
C
      UNITS(3)='DEGREE'
      UNITS(4)='S     '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-37
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,2
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
    2 CONTINUE
      IF(ICODE.NE.ENTRY(1,3)) GO TO 3
      IF(UNICDE.GE.0) VALUE=ATAN(VALUE/1000.)/.0174533
      IF(UNICDE.LT.0) VALUE=TAN(VALUE*.0174533)*1000.
```

```
      UNITS(1)=ENTRY(2,3)
      UNITS(2)=ENTRY(3,3)
      RETURN
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
    3 UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE ',I3,' NOT IN ANGLES TABLE.')
      RETURN
      END
```

```
      SUBROUTINE ASCHPS(STABFR,TYPE)
      IMPLICIT INTEGER(A-Z)
C
C
C
      COMMON /ASTAIN/SANO,SANKUN,SAILAT,SAILON,SAIDAT,SAFLAT,SAFLON,
     .               SAFDAT,SASTAT(2),PNXTAS,PENVS,SAIFLG
C
C
C
      DIMENSION STABFR(1)
C
C     CLEAN OUT STABFR(2-18).
C
      DO 1 I=2,19
    1 STABFR(I)=0
C
C
      FLD(15,3,STABFR(1))=TYPE
      FLD(0,18,STABFR(2))=SANO
      FLD(18,18,STABFR(2))=SANKUN
C
C
      CALL PKAREA(' ',SAILAT,SAILON,SAIDAT,0,' ',
     .            SAFLAT,SAFLON,SAFDAT,0,' ',STABFR(3))
C
C
      STABFR(7)=SASTAT(1)
      STABFR(8)=SASTAT(2)
C
C
      STABFR(9)=PNXTAS
C
C     PACK POINTER TO RELEVANT ENVIRONMENTAL (ACOUSTIC) STATION.
C
      STABFR(10)=PENVS
      RETURN
      END
```

```
@FOR,SW     N.CONVRT,.CONVRT

      SUBROUTINE CONVRT(VALUE,INDXS,UNICDE,UNITS)
      INTEGER TABLE,UNICDE
C
C     SET UP TABLE TO MAP VARIABLE CODE TO PROPER TABLE NUMBER.
C
      DIMENSION TABLE(144),UNITS(4)
      DATA ANULL /05763777777777/
      DATA (TABLE(I),I=1,72 )/00401,00401,07712,07712,00018,02204,00401,
     .                        00401,00018,00401,00401,00018,00501,00018,
     .                        03808,07712,07712,00018,00401,02204,00018,
     .                        00401,00401,00018,00018,02204,00000,03808,
     .                        07712,03808,03204,02904,03804,03808,
     .                        3840/
C
      DATA(TABLE(I),I=73,144)/00000,00401,02204,00000,03808,03204,00000,
     .                        00401,02204,00000,00000,00000,00401,00401,
     .                        00401,03607,04309,02906,00000,00301,00301,
     .                        04517,04209,04209,04209,04209,08601,00501,
     .                        4440/
C
C
      UNITS(1)='UNITLE'
      UNITS(2)='SS    '
      UNITS(3)='UNITLE'
      UNITS(4)='SS    '
C
C
      IF (FLD(0,36,VALUE).EQ.FLD(0,36,ANULL)) RETURN
C
C
C
C     MAP CODES 1-72 AND 131-172 TO CONTINUOUS TABLE POSITIONS 1-144.
C
      INDEX=INDXS
      IF(INDEX.GT.100) INDEX=INDEX-28
C
C     IF INDEX IS OUT OF RANGE, RETURN VALUE.
C
      IF(INDEX.GE.1.AND.INDEX.LE.144) GO TO 101
      PRINT 102,INDXS
      UNITS(1) = 'UNITS '
      UNITS(2) = 'ERROR '
      RETURN
C
C     CALCULATE LABEL FOR APPROPRIATE UNIT-CODE TABLE SUBROUTINE.
C
  101 IF(TABLE(INDEX).EQ.0) RETURN
      ITMPCD = TABLE (INDEX) / 100
      INDEX = TABLE(INDEX) - ITMPCD* 100
      IF (UNICDE .EQ. 0) UNICDE = - ITMPCD
C
C     BRANCH TO CORRECT UNIT-CODE TABLE SUBROUTINE.
C
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,101),INDEX
```

```
      C
      C     'LINEAR' TABLE.
      C
    1 CALL LINEAR(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'PLANAR' TABLE.
      C
    2 CALL PLANAR(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'VOLUME' TABLE.
      C
    3 CALL VOLUME(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'SHORT TIME' TABLE.
      C
    4 CALL SHRTME(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'LONG TIME' TABLE.
      C
    5 CALL LNGTME(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'VELOCITY' TABLE.
      C
    6 CALL SPEEDS(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'TEMPERATURE' TABLE.
      C
    7 CALL TMPTUR(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'ANGLE' TABLE.
      C
    8 CALL ANGLES(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'PER CENT' TABLE.
      C
    9 CALL PERCNT(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'PRESSURE' TABLE.
      C
   10 CALL PRESSR(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'ACOUSTIC PRESSURE' TABLE.
      C
   11 CALL ACPRES(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'FREQUENCY' TABLE.
```

```
C
   12 CALL CYCLES(VALUE,UNICDE,UNITS)
      RETURN
C
C     'ENERGY' TABLE.
C
   13 CALL ENERGY(VALUE,UNICDE,UNITS)
      RETURN
C
C     'POWER' TABLE.
C
   14 CALL POWER(VALUE,UNICDE,UNITS)
      RETURN
C
C     'INTENSITY' TABLE.
C
   15 CALL INTENS(VALUE,UNICDE,UNITS)
      RETURN
C
C     'PRESSURE LEVEL' TABLE.
C
   16 CALL PRSLVL(VALUE,UNICDE,UNITS)
      RETURN
C
C     'DENSITY' TABLE.
C
   17 CALL DENSTY(VALUE,UNICDE,UNITS)
      RETURN
C
C     'DEEBEE' TABLE.
C
   18 CALL DEEBLE(VALUE,UNICDE,UNITS)
      RETURN
  102 FORMAT(' ERROR--VARIABLE NAME CODE',I3,' OUT OF RANGE.')
      END
```

```
WELT,L      N.COPAVA,.COPAVA

      SUBROUTINE COPAVA (CRDTYP,SEQ,CARDIN)
C
C
C         DATA IS READ AND STORED INTO THE APPROPRIATE SPOTS IN COMMON
C         CPV.
C
C                                                    3/12/74   GFA
C
C
      IMPLICIT INTEGER (A-Z)
C
      COMMON /CPV   /NCON,CON(9,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
     *              NDATS,NROWS(50),NDATA,VALRNG(2,30,50),DATA(30000)
      DIMENSION ICON(9,30),IPAR(30,30),IVLRNG(2,30,50)
      EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
     *            (VALRNG(1,1,1),IVLRNG(1,1,1))
      REAL CON,PAR,DATA,VALRNG
C
C
      DIMENSION CARDIN(19),FIELD(2),UNITS(9)
C
      DATA BLNK1/'      '/
      DATA BLNK2/'    '/
      DATA NULL /07777777777776/
      DATA B1/'A     '/
      DATA B2/'   B'/
      DATA FIELD(1) /'      '/
      DATA FIELD(2) /'      '/
C
C
C
C
C
C
C
C         INITIALIZE
C
C
C
      IF (SEQ .NE. '20') GO TO 30
C
C       --PROCESS CONSTANTS--
C
C               DECODE NO. OF CONSTANTS
C
      DECODE (20,CARDIN) NCON
   20 FORMAT (T19,I3)
C           CHECK FOR ERROR IN NCON
C
C
      IF (NCON .GT. 0) GO TO 18
      WRITE (6,17) (CARDIN(I),I=1,19)
   17 FORMAT (' ERROR -- NCON .LE. 0',/1X,13A6,A2)
      STOP
C
```

```
C               CHECK FOR CONTINUATION
C
   18 IF (NCON .GT. 4) GO TO 25
C
C
      DECODE (21,CARDIN) ((CON(I,J),I=1,4),J=1,NCON)
   21 FORMAT (T17,I3,T27,I3,T20,F7.0,T30,2I3,T43,I3,T36,F7.0,
     *        T46,2I3,T59,I3,T52,F7.0,T62,2I3,T75,I3,T68,F7.0,
     *        T78,I3)
      GO TO 24
C
C
   25 DECODE (21,CARDIN) ((CON(I,J),I=1,4),J=1,4)
      READ (9,21)          ((CON(I,J),I=1,4),J=5,NCON)
   24 CALL CRDRED (CARDIN,SEQ,CRDTYP)
C
C
C          CHECK FOR PARAMETERS
C
   30 IF (SEQ .NE. '30') GO TO 40
C
C        --PROCESS PARAMETERS--
C
C               INCREMENT NO. OF PARAMETERS COUNTER
C
      NPAR = NPAR +1
C
C               DECODE NAME,UNIT,PROCESS & NO. OF VALUES
C
      DECODE (32,CARDIN) (PAR(I,NPAR),I=1,4)
   32 FORMAT (T19,4I3)
C
C
      NVAL = IPAR(1,NPAR)
      IF (NVAL .GT. 6) GO TO 35
C
C               NO, DO THIS CARD
C
      J= NVAL +4
      DECODE (33,CARDIN) (PAR(I,NPAR),I=5,J)
   33 FORMAT (T27,6F9.0)
      GO TO 37
C
C
   35 DECODE (33,CARDIN) (PAR(I,NPAR),I=5,10)
      READ (9,33) (PAR(I,NPAR),I=11,J)
C
C               READ NEXT CARD
C
   37 CALL CRDRED (CARDIN,SEQ,CRDTYP)
C
C               CONTINUE TO PROCESS ALL PARAMETERS
C
      GO TO 30
C
C
C
```

```
      40 IF (SEQ .EQ. '4C') GO TO 45
C
C          NO VARIABLES, CHECK FOR E OR A TYPE
C
          IF (FLD(C,6,CARDIN(1)) .EQ. 'E') RETURN
C
C          A TYPE, WRITE ERROR
C
          WRITE (6,41)(CARDIN(I),I=1,19)
       41 FORMAT (//,' ERROR -- NO AR4D CARD ENCOUNTERED WHEN EXPECTED.',/,
     .          1X,13A6,A2)
          STOP
C
C         --PROCESS VARIABLES--
C
C                DECODE # OF VARIABLES
C
       45 DECODE (20,CARDIN) NVAR
C
C
          IF (NVAR .GT. 7) GO TO 47
C
C                NO CONT., DO THIS CARD
C
          DECODE (46,CARDIN) (((IVAR(I,J),I=1,3),J=1,NVAR)
       46 FORMAT (T10,7(3I3))
          GO TO 50
C
C
       47 DECODE (46,CARDIN) (((IVAR(I,J),I=1,3),J=1,7)
          READ (9,48) (((IVAR(I,J),I=1,3),J=8,NVAR)
       48 FORMAT (T27,6(3I3))
C
C                READ NEXT CARD
C
       50 CALL CRDRED (CARDIN,SEQ,CRDTYP)
C
C          CHECK FOR DATA SET CARD
C
          IF (SEQ .EQ. '5D') GO TO 55
C
C          WRITE ERROR
C
          WRITE (6,52) (CARDIN(I),I=1,19)
       52 FORMAT(//,' ERROR -- AR OR ER 5J CARD NOT ENCOUNTERED WHEN EXPECTE
     .D.',/,1X,13A6,A2)
C
          STOP
C
C         --PROCESS DATA SETS CARD(S)--
C
C
C                DECODE NUMBER OF DATA SETS
C
       55 DECODE (20,CARDIN) NDATS
C
C
          NDATSC =1
```

```
      IF (NPAR .EQ. 0) GO TO 60
      DO 57 I=1,NPAR
      NDATSC = NDATSC * IPART(I)
   57 CONTINUE
C
C                 CHECK BOTH THE SAME
C
   60 IF (NDATS .EQ. NDATSC) GO TO 65
C
C                 WRITE ERROR
C
      WRITE (6,61) NDATS,NDATSC,(CARDIN(I),I=1,14)
   61 FORMAT (' ERROR -- THE NUMBER OF DATA SETS (',I3,') COUNTED DID NO
     *T AGREE W',/,'TH',I1,'THE NUMBER OF DATA SETS (',I3,') COMPUTED.',/,
     *1X,13A6,A2)
C
      STOP
C
C
   65 IF (NDATS .GT. 16) GO TO 70
C
C                 DECODE THIS CARD
C
      DECODE (66,CARDIN) (NROWS(I),I=1,NDATS)
   66 FORMAT (I7,16I4)
      GO TO 75
C
C
   70 DECODE (66,CARDIN) (NROWS(I),I=1,16)
      READ (9,66) (NROWS(I),I=17,NDATS)
C
C
   75 DO 77 I=1,NDATS
      IF (NROWS(I) .GT. 0) GO TO 77
      WRITE (6,76) I,NROWS(I),(CARDIN(J),J=1,14)
   76 FORMAT (' ERROR -- NROWS(',I2,') .LT. 1',I10,/,1X,13A6,A2)
      STOP
   77 CONTINUE
C
C
C
   80 NDATA = 0
C
C                 PROCESS ALL DATA SETS
C
      DO 100 L=1,NDATS
C
C                 COMPUTE NO. REMAINING & INIT COUNTER
C
      NOREM = NROWS(L) * NVAR
      NCONT = 0
C
C
   85 CALL MOVINT(CARDIN,21,10)
C
C                 READ A CARD
C
```

```
      CALL CRDRED (CARDIN,SEQ,CRDTYP)
C
C
      NONC = 0
C
C
      NDL = 6
      IF (NOREM .LT. 6) NDL = NOREM
C
C               MOVE NEXT SET OF CHARACTERS
C
   90 CALL MOVCHR(FIELD(1))
C
C
      NONC = NONC +1
      NCONT = NCONT +1
      IF (.NOT. (FIELD(1) .EQ. BLNK1 .AND. FIELD(2) .EQ. BLNK2))
     .   GO TO 95
C
C               BLANK DATA FIELD
C
      WRITE (6,92) NONC,(CARDIN(I),I=1,14)
   92 FORMAT (//,' ERROR-- BLANK DATA FIELD (',I1,') ENCOUNTERED.',/,
     .          1X,13A6,A2)
      STOP
C
C               CHECK FOR 'NULL' DATA FIELD
C
   95 NDATA = NDATA +1
      DATA(NDATA) = NULL
C
      IF (FIELD(1) .EQ. B1 .AND. FIELD(2) .EQ. B2) GO TO 98
C
C               DECODE NUMERIC VALUE
C
      DECODE (97,FIELD) DATA(NDATA)
   97 FORMAT (F10.0)
C
C
C
      LL = MOD(NCONT,NVAR)
      IF (LL .EQ. 0) LL = NVAR
      CALL CONVRT (DATA(NDATA),IVAR(1,LL),IVAR(2,LL),UNITS)
      IF (UNITS(1) .EQ. 'UNITS ') WRITE (6,96) (CARDIN(I),I=1,14)
   96 FORMAT (1X,13A6,A2)
C
C
C
C               CHECK FOR END OF CARD
C
   98 IF (NONC .NE. NDL) GO TO 90
C
C
      NOREM = NOREM - NDL
C
C               CHECK FOR END OF DATA SET
C
```

```
      IF (NOREM .GT. 0) GO TO 85
C
C
  100 CONTINUE
C
C
      IF (NCON .EQ. 0) GO TO 115
      DO 110 I=1,NCON
      CALL CONVRT (CON(3,I),CON(1,I),CON(2,I),UNITS)
      IF(UNITS(1).EQ.'UNITS ') WRITE(6,107) (CON(J,I),J=1,3)
  107 FORMAT(' CONSTANTS ',2I5,F21.7)
  110 CONTINUE
C
C
  115 IF (NPAR .EQ. 0 ) GO TO 150
      DO 130 I=1,NPAR
      K = IPAR(1,I)
C
           DO 120 J=1,K
           CALL CONVRT (PAR(4+J,I),IPAR(2,I),IPAR(3,I),UNITS)
      IF(UNITS(1).EQ.'UNITS ') WRITE(6,117) PAR(4+J,I),
     *IPAR(2,I),IPAR(3,I)
  117 FORMAT(' PARAM.',F21.7,2I5)
  120      CONTINUE
C
  130 CONTINUE
C
  150 RETURN
C
C         ALL DONE
C
      END
```

```
@FOR,SA     N.CRDRED,.CRDRED

      SUBROUTINE CRDRED (CARDIN,SEQ,CRDTYP)
C
C        CRDRED (CARD READ) READS 80 COLUMNS OF THE NEXT CARD AND
C        PUTS IT IN CARDIN. IT ALSO STORES COL 162 IN CRDTYP & 263
C        IN SEQ.
C
      DIMENSION CARDIN(1)
C
      READ (9,10) (CARDIN(I),I=1,14)
   10 FORMAT (13A6,A2)
C
      FLD(0,12,SEQ) = FLD(6,12,CARDIN(1))
      FLD(0,12,CRDTYP) = FLD(0,12,CARDIN(1))
C
      RETURN
      END
```

```
@FOR,S=    N.CRUINT,.CRUINT

      SUBROUTINE CRUINT
C
C         CRUINT (CRUISE INITIALIZE) INITIALIZES THE COMMON
C         /CRUISE/ EVERY TIME A NEW CRUISE RECORD IS ENCOUNTERED.
C
C
      IMPLICIT INTEGER (A-Z)
C
C
      COMMON /CRUISE/CRUNO,PNXTC,CAILAT,CAILON,CAIDAT,CAFLAT,CAFLON,
     .               CAFDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,
     .               CASTAT(2),CESTAT(2),PACS,PENS,NCARDS,NOTES(14,132),
     .               CAFLG
C
C
C
      PNXTC = 0
      CAFLG = 0
      CAILON = -179600
      CAIDAT = 991232
      CAFLAT = 90000
      CAFLON = 179600
      CAFDAT = 0
C
      CEILAT = -90000
      CEILON = -179600
      CEIDAT = 991232
      CEFLAT = 90000
      CEFLON = 179600
      CEFDAT = 0
C
C
      RETURN
      END
```

```
@ELT,L     N,CRUSUM,,CRUSUM

      SUBROUTINE CRUSUM (BFR1,BFR2)
C
C
C
      IMPLICIT INTEGER (A-Z)
C
C
      COMMON /CRUISE/CRUNO,PNXTC,CAILAT,CAILON,CAIDAT,CAFLAT,CAFLON,
     *               CAFDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,
     *               CASTAT(2),CESTAT(2),PACS,PENS,NCARDS,NOTES(14,132),
     *               CAFLG
C
C
C
      COMMON /ASTATN/SANO,SANRUN,SAILAT,SAILON,SAIDAT,SAFLAT,SAFLON,
     *               SAFDAT,SASTAT(2),PNXTAS,PENVS,SAIFLG
C
C
C
C
      COMMON /SECTO / ID(7),IDATE,NXTSEC,LSTADR,LEXP,NOCRU,NOSTA,
     *                LSTADC,LSTADA,LSTADE
C
C
C
      FLD(18,18,SECADR) = FLD(0,18,LSTADC)
      FLD(18,18,MSEC)   = FLD(18,18,LSTADC)
C
      CALL DSKRED (SECADR,MSEC,BFR1)
C
C         UNPACK IT
C
      CALL UNPKCR (BFR1)
C
C
      FLD(18,18,SECAD1) = FLD( 0,18,LSTADA)
      FLD(18,18,MSEC1)  = FLD(18,18,LSTADA)
C
      CALL DSKRED (SECAD1,MSEC1,BFR2)
C
C         UNPACK IT
C
      CALL ASUNPK (BFR2,TYPE)
C
      IF (SAILAT .GT. CAILAT) CAILAT = SAILAT
      IF (SAFLAT .GT. CAILAT) CAILAT = SAFLAT
      IF (SAILAT .LT. CAFLAT) CAFLAT = SAILAT
      IF (SAFLAT .LT. CAFLAT) CAFLAT = SAFLAT
C
      IF (CAFLG .NE. 0) GO TO 50
      CAILON = EAST(SAILON,SAFLON)
      CAFLON = WEST(SAILON,SAFLON)
      GO TO 60
```

```
      50 CAILON = EAST(SAILON,CAILON)
         CAFLON = WEST(SAFLON,CAFLON)
C
      60 IF (SAIDAT .LT. CAIDAT) CAIDAT = SAIDAT
         IF (SAFDAT .GT. CAFDAT) CAFDAT = SAFDAT
C
         CASTAT(1) = OR(CASTAT(1),SASTAT(1))
         CASTAT(2) = OR(CASTAT(2),SASTAT(2))
C
         FLD(18,18,SECADI) = FLD( 0,18,LSTADE)
         FLD(18,18,MSECI)  = FLD(18,18,LSTADE)
C
         CALL DSKRED (SECADI,MSECI,BFR2)
C
C           UNPACK IT
C
         CALL ASUNPK (BFR2,TYPE)
C
         IF (SAILAT .GT. CEILAT) CEILAT = SAILAT
         IF (SAFLAT .GT. CEILAT) CEILAT = SAFLAT
         IF (SAILAT .LT. CEFLAT) CEFLAT = SAILAT
         IF (SAFLAT .LT. CEFLAT) CEFLAT = SAFLAT
C
         IF (CAFLG .NE. 0) GO TO 100
         CAFLG = 1
         CEILON =EAST(SAILON,SAFLON)
         CEFLON = WEST(SAILON,SAFLON)
         GO TO 110
     100 CEILON = EAST(SAILON,CEILON)
         CEFLON = WEST(SAFLON,CEFLON)
C
     110 IF (SAIDAT .LT. CEIDAT) CEIDAT = SAIDAT
         IF (SAFDAT .GT. CEFDAT) CEFDAT = SAFDAT
C
         CESTAT(1) = OR(CESTAT(1),SASTAT(1))
         CESTAT(2) = OR(CESTAT(2),SASTAT(2))
C
C           REPACK CRUISE
C
         CALL PCKCRU (BFRI,N)
C
C
         CALL DSKRIT (SECADR,MSEC,BFRI)
C
C           DONE
C
         RETURN
         END
```

```
@FOR,Sn     N.CYCLES,.CYCLES

      SUBROUTINE CYCLES(VALUE,UNICDE,UNITS)
C
C     TABLE OF CYCLES MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA ((ENTRY(I,J),I=1,3),J=77,78,79/
C
C     PUT IN ALPHA UNITS.
C
      DATA (((ENTRY(I,J),I=2,3),J=1,3)/'HERTZ ','      ',
     *              'KILOHE','RTZ   ','C.P.M.','      '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='HERTZ '
      UNITS(4)='      '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-77
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,3)/1.000,1.003,6.001/
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C        CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,3
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
    2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
```

```
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
101   FORMAT(' ERROR--UNIT CODE',I3,' NOT IN FREQUENCY TABLE.')
      RETURN
      END
```

```
@FOR,Sn     N.DBLOSS,.DBLOSS

      SUBROUTINE DBLOSS(VALUE,UNICDE,UNITS)
C
C     TABLE FOR DB LOSS AND ATTENUATION CONVERSION FACTORS.
C
      DIMENSION FACTOR(8)
      INTEGER  ENTRY(3,8),UNITS(4),UNICDE
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,8)/87,81,82,90,92,93,94,95/
C
C     ENTER CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,8)/.8,-59.2,0.,-60.,0.,-.8,59.2,60./
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,8)/'DB//1 ','YARD  ',
     *            'DB//1 ','KYARD ','DB//1 ','METER ',
     *            'DB//1 ','KMETER','DB/MET','ER    ',
     *            'DB/YAR','D     ','DB/KYA','RD    ',
     *            'DB/KME','TER   '/
C
C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
C
C     RETRIEVAL IN STANDARD DB LOSS UNITS.
C
      IF(UNICDE.NE.0) GO TO 1
      UNICDE=-82
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='DB//1'
      UNITS(4)='METER '
      GO TO 2
C
C     RETRIEVAL IN STANDARD ATTENUATION UNITS.
C
    1 IF(UNICDE.NE.-1) GO TO 2
      UNICDE=-92
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='DB/MET'
      UNITS(4)='ER    '
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
    2 ICODE=IABS(UNICDE)
      DO 3 I=1,8
```

```
      IF(ICODE.NE.ENTRY(1,I)) GO TO 3
      IF(UNICDE.GT.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
    3 CONTINUE
C
C     ELSE, UNIT CODE IS NOT CONVERTIBLE.
C
      PRINT 101,ICODE
      PRINT 112
  112 FORMAT(' ERROR--UNIT CODE NOT CONVERTIBLE!')
      UNITS(1)='UNITS '
      RETURN
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN DB LOSS TABLE.')
      END
```

```
@FOR,S A.DEEBLE,.DEEBEE

      1.            SUBROUTINE DEEBEE(VALUE,UNICDE,UNITS)
      2.      C
      3.      C     TABLE OF DB UNITS.
      4.      C
      5.            INTEGER ENTRY(3,27),UNITS(4),UNICDE
      6.      C
      7.      C     PUT CODES INTO 'ENTRY'.
      8.      C
      9.            DATA (ENTRY(I,1),I=1,27)/39,40,41,58,66,67,68,69,72,73,83,86,96,
     10.           * 87,81,82,90,92,93,94,95,64,65,91,101,102,103/
     11.      C
     12.      C     ENTER ALPHA UNITS.
     13.      C
     14.            DATA ((ENTRY(I,J),I=2,3),J=1,27)/'DB/YD**','2     ',
     15.           *                 '//1UBR','2/S/HZ','CM3/6R','AM    ',
     16.           *                 'PRESSR',' RATIO','DB//1W','ATT   ',
     17.           *                 'DB//1W','/M**2 ','DB//1W','/CM**2',
     18.           *                 '//1ERG','SC/CM2','DB//1E','RG/CM2',
     19.           *                 '1ERG/S','EC/CM ','///JLE/','CM2-HZ',
     20.           *                 'DB   ',',     ','///ERG/','CM2BIT',
     21.           *                 'DB   ',',     ','DB//1 ','KYARD ',
     22.           *                 'DB//1 ','METER ','DB//1 ','KMETER',
     23.           *                 'DB/MET','ER    ','DB/YAR','D     ',
     24.           *                 'DB/KYA','RD    ','DB/KME','TER   ',
     25.           *                 'DB//1U','BARQIM','DB//1U','BARWIT',
     26.           *                 'DB//1U','PAQIM ','DB//1U','PAQITD',
     27.           *                 '//1UBA','RZSQIT','///1UPA','2 SQIM'/
     28.      C
     29.      C     IF UNICDE.EQ.0, THEN SET UNITS TO "DB AS MEAS.".
     30.      C
     31.            UNITS(1)='DB AS '
     32.            UNITS(2)='MEAS. '
     33.      C
     34.      C     SET INDICATOR FOR CASE UNPACK IS BYPASSED.
     35.      C
     36.            ITST=0
     37.      C
     38.      C     IF RETRIEVING, UNPACK VALUE AND UNIT CODE.
     39.      C
     40.            IF(UNICDE.GE.1) GO TO 1
     41.            ICODE=VALUE
     42.            IF(UNICDE.EQ.0.AND.ICODE.EQ.-99999999) RETURN
     43.            IF(UNICDE.NE.0.AND.ICODE.EQ.-99999999) GOTO 5
     44.            UNICDE=0
     45.            UNICDE=IABS(FLD(29,7,VALUE))
     46.      C
     47.      C     IF WE AREN'T RETRIEVING, THEN RETURN.
     48.      C
     49.            IF(ICODE.EQ.0.AND.UNICDE.EQ.0) RETURN
     50.      C
     51.      C     SET INDICATOR FOR CASE PACK IS BYPASSED.
     52.      C
     53.          5 ITST=1
     54.      C
```

```
      C     TEST TO SEE IF CODE IS IN TABLE.
      C
        1 ICODE=IABS(UNITCODE)
          DO 2 I=1,27
          IF(ICODE.EQ.ENTRY(1,I)) GO TO 3
        2 CONTINUE
      C
      C     IF CODE NOT FOUND, WRITE ERROR.
      C
          PRINT 101,ICODE
          PRINT 102,UNICODE,VALUE,ITST
      102 FORMAT(' UNICODE=',I6,'  VALUE(OCTAL)=',O12,'  ITST=',I5)
          UNITS(1) = 'UNITS '
          UNITS(2) = 'ERROR '
          RETURN
      C
      C     PACK CODE AND VALUE, IF NECESSARY.
      C
        3 IF(ITST.EQ.1) GO TO 4
          FLD(29,7,VALUE)=FLD(29,7,UNICODE)
      C
      C     SET UNITS, RETURN.
      C
        4 UNITS(1)=ENTRY(2,I)
          UNITS(2)=ENTRY(3,I)
          UNITS(3)=ENTRY(2,I)
          UNITS(4)=ENTRY(3,I)
          RETURN
      101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN DEEBEE TABLE.')
          END
```

```
@FOR,SW     N.DENSTY,.DENSTY

      SUBROUTINE DENSTY(VALUE,UNICDE,UNITS)
C
C     TABLE OF DENSITY MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C.
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(I,J),I=1,3)/95,96,97/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,3)/'GMS/CM','**3  ',
     *            'KG/MET','ER**3 ','LB/FT**','3    '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='GMS/CM'
      UNITS(4)='**3    '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-90
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,3)/1.000,1.003,7.769223D4/
C
C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,3
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
    2 CONTINUE
```

```
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      PRINT 101,ICODE
101   FORMAT(' ERROR--UNIT CODE',I3,' NOT IN DENSITY TABLE.')
      RETURN
      END
```

```
@ELT,L      N.DSKRED,.DSKRED

      SUBROUTINE DSKRED(IN,IS,ADR)
C
C
      CALL SETADR(10,IN)
      IW = IS * 28
      CALL NTRAN(10,2,IW,ADR,L)
      CALL NTRAN(10,22)
C
      IF (L .GT. 0) RETURN
C
      WRITE (6,10) IN,IS,L
   10 FORMAT(//,' ERROR IN DSKRED',3I10)
      RETURN 0
      END
```

```
@ELT,L      N.DSKRIT,.DSKRIT

      SUBROUTINE DSKRIT(IN,IS,ADR)
C
C
      CALL SETADR(IJ,IN)
      I# = IS * 2d
      CALL NTRAN(IC,I,I@,ADR,L)
      CALL NTRAN(IO,22)
C
      IF (L .GT. 0) RETURN
C
      WRITE (6,10) IN,IS,L
   10 FORMAT(//,' ERROR IN DSKRIT',3I13)
      RETURN 0
      END
```

```
WELT,L      N.EAST,.EAST

      1.            INTEGER FUNCTION EAST (A,B)
      2.      C
      3.      C
      4.            INTEGER A,B
      5.      C
      6.            IF ((ABS(A-B) .GT. 180000) GO TO 10
      7.            EAST = MAX(A,B)
      8.            RETURN
      9.      C
     10.         10 EAST = B
     11.            IF (A .LE. 0) EAST = A
     12.            RETURN
     13.            END
```

```
@FOR,SA     N.ENERGY,,ENERGY

      SUBROUTINE ENERGY(VALUE,UNICDE,UNITS)
C
C     TABLE OF ENERGY MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,5),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(5)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,5)/61,62,63,88,89/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,5)/'ERGS  ','      ',
     *              'JOULES','      ','FT.-LB','S.    ',
     *              'KGM ME','TERS  ','&HBTU'S ','      '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='KGM ME'
      UNITS(4)='TERS  '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-88
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,5)/1.0197D-8,.10197D0,.13825550D0,100,107.500/
C
C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,5
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
```

```
      C
    2 CONTINUE
      C
      C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
      C
          PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN ENERGY TABLE.')
          RETURN
          END
```

```
GELT,C       N.ENSTAT,.ENSTAT

      SUBROUTINE ENSTAT (STATUS)
C
      IMPLICIT INTEGER (A-Z)
C
C
      COMMON /CPV    /NCON,CON(4,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
     *                NDATS,NROWS(50),NDATA,VALRNG(2,30,50),DATA(30000)
      DIMENSION ICON(4,30),IPAR(30,30),IVLRNG(2,30,50)
      EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
     *            (VALRNG(1,1,1),IVLRNG(1,1,1))
      REAL CON,PAR,DATA,VALRNG
C
C
      DIMENSION STATUS(2)
C
C          CHECK FOR CONSTANTS
C
      IF (NCON .EQ. 0) GO TO 50
C
      DO 40 I=1,NCON
C
      IF (ICON(1,I) .LT. 101) GO TO 7
      ICOD = IABS(ICON(1,I) - 101)
C
      ISUB = 1
      IF (ICOD .LE. 35) GO TO 10
      ISUB = 2
      ICOD = IABS(ICOD -36)
C
      IF (ICOD .LE. 35) GO TO 10
    7 WRITE (6,4) ICON(1,I)
    4 FORMAT (' ERROR -- ENSTAT - ILLEGAL CODE',I10)
      RETURN 0
C
   10 FLD(ICOD,1,STATUS(ISUB)) = 1
   40 CONTINUE
C
C          CHECK PARAMETERS
C
   50 IF (NPAR .EQ. 0) GO TO 70
C
      DO 65 I=1,NPAR
C
      IF (IPAR(2,I) .LT. 101) GO TO 57
      ICOD = IABS(IPAR(2,I) -101)
C
      ISUB = 1
      IF (ICOD .LE. 35) GO TO 60
      ISUB = 2
      ICOD = IABS(ICOD -36)
C
      IF (ICOD .LT. 35) GO TO 60
   57 WRITE (6,4) IPAR(2,I)
      RETURN 0
```

```
      C
   60 FLD(ICOD,I,STATUS(ISUB)) = 1
   65 CONTINUE
      C
      C          CHECK VARIABLES
      C
   70 IF (NVAR .EQ. 0) GO TO 100
      C
         DO 90 I=1,NVAR
      C
         IF (IVAR(1,I) .LT. 101) GO TO 77
         ICOD = IABS(IVAR(1,I) - 101)
      C
         ISUB = 1
         IF (ICOD .LE. 35) GO TO 60
         ISUB = 2
         ICOD = IABS(ICOD -36)
      C
         IF (ICOD .LT. 35) GO TO 80
   77 WRITE (6,9) IVAR(1,I)
    9    RETURN 0
      C
   80 FLD(ICOD,I,STATUS(ISUB)) = 1
   90 CONTINUE
      C
  100 RETURN
         END
```

```
@ELT,L      N,EXPINT,,EXPINT

      SUBROUTINE EXPINT
C
      IMPLICIT INTEGER (A-Z)
C
      COMMON /EXPMNT/EXNO,PNXTE,PCRU,EAILAT,EAILON,EAIDAT,EAFLAT,EAFLON,
     *               EAFDAT,EEILAT,EEILON,EEIDAT,EEFLAT,EEFLON,EEFDAT,
     *               EASTAT(2),EESTAT(2),EAFLG
C
C
C
      PNXTE = 0
      PCRU  = 0
      EAILAT = -90000
      EAFLG = 0
      EAIDAT = 991232
      EAFLAT = 90000
      EAFLON = 179600
      EAFDAT =  0
C
      EEILAT = -90000
      EEILON = -179600
      EEIDAT = 991232
      EEFLAT = 90000
      EEFLON = 179600
      EEFDAT = 0
C
      RETURN
      END
```

```
@ELT,L     N.EXPSUM,.EXPSUM

      1.            SUBROUTINE EXPSUM (BFR1,BFR2)
      2.      C
      3.      C
      4.      C
      5.            IMPLICIT INTEGER (A-Z)
      6.      C
      7.      C
      8.      C
      9.            COMMON /EXPMNT/EXNO,PNXTE,PCRU,EAILAT,EAILON,EAIDAT,EAFLAT,EAFLON,
     10.           *               EAFDAT,EEILAT,EEILON,EEIDAT,EEFLAT,EEFLON,EEFDAT,
     11.           *               EASTAT(2),EESTAT(2),EAFLG
     12.      C
     13.      C
     14.            COMMON /CRUISE/CRUNO,PNXTC,CAILAT,CAILON,CAIDAT,CAFLAT,CAFLON,
     15.           *               CAFDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,
     16.           *               CASTAT(2),CESTAT(2),PACS,PENS,NCARDS,NOTES(14,132),
     17.           *               CAFLG
     18.      C
     19.      C
     20.            COMMON /SECTO /ID(7),IDATE,NXTSEC,LSTADR,LEXP,NOCRU,NOSTA,
     21.           *               LSTADC,LSTADA,LSTADE
     22.      C
     23.            SECADR = FLD( 0,18,LSTADR)
     24.            MSEC = FLD(18,18,LSTADR)
     25.      C
     26.            CALL DSKRED (SECADR,MSEC,BFR1)
     27.      C
     28.      C           UNPACK IT
     29.      C
     30.            CALL UNPKXP(BFR1)
     31.      C
     32.            SECADI = FLD( 0,18,LSTADC)
     33.            MSECI =  FLD(18,18,LSTADC)
     34.      C
     35.      C
     36.            CALL DSKRED (SECADI,MSECI,BFR2)
     37.      C
     38.      C           UNPACK IT
     39.      C
     40.            CALL UNPKCR(BFR2)
     41.      C
     42.            IF (CAILAT .GT. EAILAT) EAILAT = CAILAT
     43.            IF (CAFLAT .GT. EAILAT) EAILAT = CAFLAT
     44.            IF (CAILAT .LT. EAFLAT) EAFLAT = CAILAT
     45.            IF (CAFLAT .LT. EAFLAT) EAFLAT = CAFLAT
     46.      C
     47.            IF (EAFLG .NE. 0) GO TO 50
     48.            EAILON = EAST(CAILON,CAFLON)
     49.            EAFLON = WEST (CAILON,CAFLON)
     50.            GO TO 60
     51.         50 EAILON = EAST(CAILON,EAILON)
     52.            EAFLON = WEST(CAFLON,EAFLON)
     53.      C
     54.         60 IF (CAIDAT .LT. EAIDAT) EAIDAT = CAIDAT
```

```
      IF (CAFDAT .GT. EAFDAT) EAFDAT = CAFDAT
C
      EASTAT(1) = OR(EASTAT(1),CASTAT(1))
      EASTAT(2) = OR(EASTAT(2),CASTAT(2))
C
      IF (CEILAT .GT. EEILAT) EEILAT = CEILAT
      IF (CEFLAT .GT. EEILAT) EEILAT = CEFLAT
      IF (CEILAT .LT. EEFLAT) EEFLAT = CEILAT
      IF (CEFLAT .LT. EEFLAT) EEFLAT = CEFLAT
C
      IF (EAFLG .NE. 0) GO TO 100
      EAFLG = 1
      EAILON = EAST (CEILON,CEFLON)
      EAFLON = WEST (CEILON,CEFLON)
      GO TO 110
  100 EEILON = EAST(CEILON,EEILON)
      EEFLON = WEST(CEFLON,EEFLON)
C
  110 IF (CEIDAT .LT. EEIDAT) EEIDAT = CEIDAT
      IF (CEFDAT .GT. EEFDAT) EEFDAT = CEFDAT
C
      EESTAT(1) = OR(EESTAT(1),CESTAT(1))
      EESTAT(2) = OR(EESTAT(2),CESTAT(2))
C
C           REPACK EXPERIMENT
C
      CALL PCKEXP (BFRI,N)
C
C           WRITE EXPERIMENT RECORD BACK
C
      CALL DSKRIT (SECADR,NSEC,BFRI)
C
C           DONE !
C
      RETURN
      END
```

```
@ELT,L     N.INITD..INITD

      C
      C
            DIMENSION D(28)
            INTEGER D
            D( 1) = '   NAV'
            D( 2) = 'SEA UN'
            D( 3) = 'DER@AT'
            D( 4) = 'ER ACO'
            D( 5) = 'USTIC '
            D( 6) = 'DATA B'
            D( 7) = 'ANK   '
            D( 8) =  32674
            D( 9) =      1
            D(10) =      0
            D(11) =      0
            D(12) =      0
            D(13) =      0
            D(14) = 0
            D(15) = 0
            D(16) = 0
      C
            CALL NTRAN (10,22)
            CALL SETADR(10,D)
            CALL NTRAN(10,1,28,D,LSTAT)
            CALL NTRAN(10,22)
      C
            IF (LSTAT .GT. 0) STOP
      C
            WRITE (6,10) LSTAT
         10 FORMAT (' ERROR -- ',I5)
            STOP
            END
```

```
@FOR,SR     N,INTENS,,INTENS

      SUBROUTINE INTENS(VALUE,UNICDE,UNITS)
C
C     TABLE OF INTENSITY LEVEL CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,3)/74,75,76/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,3)/'WATTS/','M**2  ',
     *            'WATTS/','CM**2 ','WTS/CM','**2/HZ'/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='WATTS/'
      UNITS(4)='M**2  '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-74
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,3)/100.,1.0D4,1.0D4/
C
C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,3
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
```

```
      2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN INTENSITY TABLE.')
      RETURN
      END
```

```
@FOR,S#     N.LINEAR.LINEAR

      SUBROUTINE LINEAR(VALUE,UNICDE,UNITS)
C
C     TABLE OF LINEAR MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,12),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(12)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,12)/2,3,4,5,6,7,8,9,10,11,12,86/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,12)/'MILLIM','ETERS ',
     *                 'CENTIM','ETERS ','METERS',
     *                 'KILOME','TERS  ','INCHES',
     *                 'FEET  ','      ','YARDS ',
     *                 'KILOYA','RDS   ','NAUT. ','MILES ',
     *                 'STAT. ','MILES ','FATHOM','S     ',
     *                 'MICRON','S     '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='METERS'
      UNITS(4)='      '
C
C     IF UNICDE = 0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-4
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,12)/.001D0,.01D0,.1D0,1.0D3,2.54000D-2,
     *                        .3048D0,.9144D0,1.9440192170D2,
     *                        1.852D3,1.6093D3,1.82880D3,1.0D-6/
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,12
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
```

```
      C
      C     ELSE, CHECK REST OF TABLE.
      C
    2 CONTINUE
      C
      C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
      C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN LINEAR TABLE.')
      RETURN
      END
```

```
@FOR,S# N.LNGTHE,.LNGTME

      SUBROUTINE LNGTME(VALUE,UNICDE,UNITS)
C
C     TABLE OF LNGTME MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,2),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(2)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA ((ENTRY(I,J),I=1,2),J=1,2)/26,27/
C
C     PUT IN ALPHA UNITS.
C
      DATA (((ENTRY(I,J),I=2,3),J=1,2)/'MONTHS',' ',
     *                 'YEARS ',' '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='MO.   '
      UNITS(4)='      '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-26
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,2)/100,1200/
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,2
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
    2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
```

```
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN LONG TIME TABLE.')
      RETURN
      END
```

```
@ELT,L      N,MAIN,*MAIN

      C
      C          MAIN PROGRAM FOR BUILDING NAVDAB DATA BASE
      C
      C
      C                                                    3/26/79  GFA
      C
            IMPLICIT INTEGER (A-Z)
            PARAMETER LCRUN = 58,LCCPV = 4165
      C
      C
            COMMON/SECTG / ID(7),IDATE,NXTSEC,LSTADR,LEXP,NOCRU,NOSTA,
           *               LSTADC,LSTADA,LSTADE
            DIMENSION SECTZ(28)
            EQUIVALENCE (ID(1),SECTZ(1))
      C
      C
      C
            COMMON /EXPMNT/EXNO,PNXTE,PCRU,EAILAT,EAILON,EAIDAT,EAFLAT,EAFLON,
           *               EAFDAT,EEILAT,EEILON,EEIDAT,EEFLAT,EEFLON,EEFDAT,
           *               EASTAT(2),EESTAT(2),EAFLG
      C
      C
            COMMON /CRUISE/CRUNO,PNXTC,CAILAT,CAILON,CAIDAT,CAFLAT,CAFLON,
           *               CAFDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,
           *               CASTAT(2),CESTAT(2),PACS,PENS,NCARDS,NOTES(14,132),
           *               CAFLG
      C
      C
            COMMON /ASTATN/SANO,SANRUN,SAILAT,SAILON,SAIDAT,SAFLAT,SAFLON,
           *               SAFDAT,SASTAT(2),PNXTAS,PENVS,SAIFLG
      C
      C
      C
      C
            COMMON /RUN    /PORA,RILAT,RINS,RFLAT,RFNS,RILON,RIEW,RFLON,RFEW,
           *               RIMO,RIDAY,RIYR,RITIM,RIZON,RFMO,RFDAY,RFYR,RFTIM,
           *               RFZON,NAVCOD,DTASRC,SYSSRC,SYSRCR,SRCTYP,RCRTYP,
           *               CLAS,DESCR,#AVDIR,#AVHT,HTUNIT,#AVPRD,PRDUNT,
           *               SEASTA,#NDDIR,#NDVEL,VELUNT,S#LDIR,S#LHT,
           *               SHTUNT,S#LPRD,SPDUNT,#EATHR,BOTDPH,DPHUNT,BTMSLP,
           *               BATHY,INFO(9),RSTAT(2),RUNNO
            REAL #AVDIR,#AVHT,#AVPRD,SEASTA,#NDDIR,#NDVEL,S#LDIR,
           *     S#LHT,S#LPRD,#EATHR,BOTDPH,BTMSLP
      C
      C
            COMMON /CPV    /NCON,CON(9,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
           *               NDATS,NHO#S(50),NDATA,VALRNG(2,30,50),DATA(30000)
            DIMENSION ICON(9,30),IPAR(30,30),IVLRNG(2,30,50),IEQV(30000)
            EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
           *           (VALRNG(1,1,1),IVLRNG(1,1,1),IEQV(1))
            REAL CON,PAR,DATA,VALRNG
      C
      C
            DATA BB1   /'B  B'/
```

```
      DATA BB2 /'BB'/
      DATA BB3 /'B'/
      DATA BB4 /'BBB'/
      DIMENSION CARDIN(14),NAVID(7),STABFR(3000),TEMP(2)
      DATA NAVID   /'   NAVSEA UNDERWATER ACOUSTIC DATA BANK   '/
      DATA CRDTYP/'      '/
      DATA SEQ   /'      '/
      DATA BLNK  /'      '/
C
      DIMENSION UNITS(4)
C
      CALL NTRAN (10,22)
C
      READ(5,1) NOFILP,NDATE
    1 FORMAT()
C
    5 CALL DSKRED(0,1,SECTZ)
C
C         VERIFY CORRECT DISK
C
      DO 10 I=1,7
      IF (ID(I) .NE. NAVID(I)) GO TO 20
   10 CONTINUE
      GO TO 25
C
C         WRONG DISK
C
   20 WRITE(6,21) (ID(I),I=1,7)
   21 FORMAT (//,' WRONG DISK MOUNTED',/,1X,7A6)
      STOP
C
C         INIT NO. RUNS COUNTER AND READ A DATA CARD.
C
   25 NRUNS = 0
C
      CALL NTRAN (11,10)    REWIND 11
C
   27 CALL CRDRED (CARDIN,SEQ,CRDTYP)
C
C         CHECK FOR NEW EXPERIMENT CARD
C
      IF (CRDTYP .NE. 'NE') GO TO 50
C
      IF (NOSTA .EQ. 0 .AND. NOCRU .EQ. 0) GO TO 37
C
C               WRITE ERROR
C
      WRITE(6,32) NOCRU,NOSTA
   32 FORMAT (//,' ERROR-- NE CARD ENCOUNTERED. NOCRU=',I5,' NOSTA=',I5)
      STOP
C
   37 DECODE(34,CARDIN) IEXPN,NNOCRU
   34 FORMAT (T4,I3,I2)
C
      IF (LEXP .NE. IEXPN) GO TO 45
C
C               WRITE ERROR. DUPLICATE EXP #'S
```

```
      C
            WRITE (6,40) LEXP,IEXPN
         40 FORMAT(//,' ERROR-- TRYING TO ENTER SAME EXPERIMENT NO.',2I5)
            STOP
      C
      C              CHECK FOR NEW # OF CRUISES < 0
      C
         45 IF (NNOCRU .GT. 0) GO TO 49
      C
      C              WRITE ERROR
      C
            WRITE(6,47) NNOCRU
         47 FORMAT(//,' ERROR-- NUMBER OF CRUISES INVALID.',I5)
            STOP
      C
         49 NOCRU = NNOCRU
            LEXP = IEXPN
      C
            EXNO = LEXP
            CALL EXPINT
      C
            CALL PCKEXP(DATA,NSEC)
      C
            CALL DSKRIT(NXTSEC,NSEC,DATA)
            PRINT 48
         48 FORMAT (' NEW EXP WRITTEN')
      C
            IF (LSTADR .EQ. 0) GO TO 46
            FLD (18,18,SECADR) = FLD(0,18,LSTADR)
            FLD (18,18,MSEC  ) = FLD(18,18,LSTADR)
      C
            CALL DSKRED (SECADR,MSEC,DATA)
      C
      C          UPDATE POINTER
      C
            FLD(0,18,DATA(15)) = FLD(18,18,NXTSEC)
            FLD(18,18,DATA(15)) = FLD(18,18,NSEC)
            CALL DSKRIT (SECADR,MSEC,DATA)
      C
         46 FLD(0,18,LSTADR) = FLD(18,18,NXTSEC)
            FLD(18,18,LSTADR)= FLD(18,18,NSEC)
            NXTSEC = NXTSEC + NSEC
      C
            LSTADC = 0
      C
      C          READ NEXT CARD
      C
            GO TO 27
      C
      C
      C          CHECK FOR NEW CRUISE CARD
      C
         50 IF (CRDTYP .NE. 'NC') GO TO 130
      C
      C          CHECK STATION COUNT FOR ZERO
      C
         55 IF (NOSTA .EQ. 0) GO TO 60
```

```
C
C          WRITE ERROR
C
      WRITE (6,57) LEXP,NOCRU,NOSTA,(CARDIN(I),I=1,14)
   57 FORMAT(' ERROR--NEW CRUISE ENCOUNTERED... NOSTA NE 0',/,3110,/,
     *       14,13A6,A2)
      STOP
C
   60 DECODE (61,CARDIN) IEXPN,CRUNO,NNOSTA,NCARDS
   61 FORMAT(T4,I3,I2,I3,2X,I3)
      IF (IEXPN .EQ.LEXP) GO TO 70
C
C          WRITE ERROR
C
      WRITE (6,63) LEXP,(CARDIN(I),I=1,14)
   63 FORMAT(' ERROR-- EXPERIMENT NO. NOT SAME...', I10,/,1X,13A6,A2)
      STOP
C
C          CHECK STATN NO .LT. 0
C
   70 IF (NNOSTA .GT. 0) GO TO 75
C
C          WRITE ERROR
C
      WRITE (6,72) NNOSTA,(CARDIN(I),I=1,14)
   72 FORMAT (' ERROR-- NEW NO. STATIONS INCORRECT...',I10,/,1X,13A6,A2)
      STOP
C
   75 NOSTA = NNOSTA
C
C
C          READ IN ALL NOTE CARDS
C
      DO 90 I=1,NCARDS
      READ (9,85) (NOTES(J,I),J=1,14)
   85 FORMAT (13A6,A2)
   90 CONTINUE
C
      CALL CRUINT
C
      CALL PCKCRU(DATA,NSEC)
C
      CALL DSKRIT(NXTSEC,NSEC,DATA)
      PRINT 95
   95 FORMAT (' NEW CRUISE WRITTEN')
C
C          CHECK CRUISE # = 1
C
      IF (LSTADC .NE. 0) GO TO 110
C
      FLD(18,18,SECADR) = FLD(0,18,LSTADR)
      FLD(18,18,NSEC)   = FLD(18,18,LSTADR)
C
      CALL DSKRED (SECADR,NSEC,DATA)
C
C              UPDATE DOWN PTR.
C
```

```
      FLD( 0,18,DATA(16))= FLD(18,18,NXTSEC)
      FLD(18,18,DATA(16))= FLD(18,18,NSEC)
C
C                 WRITE IT BACK
C
      CALL DSKRIT (SECADR,MSEC,DATA)
C
      FLD(0,36,LSTADC) = FLD(0,36,DATA(16))
C
C        ZERO LAST ADDRESSES FOR ACOUSTIC AND ENV STATIONS
C
      LSTADA = 0
      LSTADE = 0
C
      NXTSEC = NXTSEC + NSEC
C
C          GO BACK & READ NEXT CARD
C
      GO TO 27
C
C
110 FLD(18,18,SECADR) = FLD( 0,18,LSTADC)
      FLD(18,18,MSEC)   = FLD(18,18,LSTADC)
C
      CALL DSKREDI(SECADR,MSEC,DATA)
C
C                 UPDATE NEXT PTR.
C
      FLD( 0,18,DATA(3)) = FLD(18,18,NXTSEC)
      FLD(18,18,DATA(3)) = FLD(18,18,NSEC)
C
C                 WRITE IT BACK
C
      CALL DSKRIT (SECADR,MSEC,DATA)
C
C                 UPDATE LAST CRUISE ADDRESS & SET NEXT AVAIL. ADDRESS
C
      FLD(0,36,LSTADC) = FLD(0,36,DATA(3))
C
      NXTSEC = NXTSEC + NSEC
C
C          GO BACK & READ NEXT CARD
C
      GO TO 27
C
C        ---- END OF PROCESSING FOR NE, NC & NT CARD TYPES ----
C
C
C          CHECK FOR ACOUSTIC RUN ('AI')
C
130 IF (CRDTYP .NE. 'AI') GO TO 176
C
C
C        --PROCESS ACOUSTIC RUN INFO--
C
C                 CHECK NOSTA.
C
```

```
      IF (NOSTA .NE. 0) GO TO 140
C
C               WRITE ERROR
C
      WRITE (6,132) (CARDIN(I),I=1,19)
  132 FORMAT (' ERROR -- NEW ACOUSTIC RUN ENCOUNTERED. NOSTA = 0.',/,1X,
     .        13A6,A2)
      STOP
C
C               CHECK FOR SEQ '10'
C
  140 IF (SEQ .EQ. '10') GO TO 146
C
C               WRITE ERROR
C
      WRITE (6,145) (CARDIN(I),I=1,19)
  145 FORMAT (' ERROR -- ACOUSTIC RUN SEQ NOT 10.',/,1X,13A6,A2)
      STOP
C
C               INCREMENT # OF RUNS COUNTER
C
  146 NRUNS = NRUNS + 1
C
C               DECODE DATA IN HEADER CARD (EXCEPT TIME)
C
      DECODE (147,CARDIN) PORA,RILAT,RINS,RFLAT,RFNS,RILON,RIEW,RFLON,
     .                    RFEW,RIMO,RIDAY,RIYR,RIZON,RFMO,RFDAY,RFYR,
     .                    RFZON,NAVCOD,DTASRC,SYSSRC,SYSMCR,SRCTYP,
     .                    RCRTYP,CLAS,DESCR,RUNNO
  147 FORMAT (T19,A1,2(I5,A1),2(I6,A1),2(3I2,4X,A1),7I2,I4,T11,I3)
C
C               EXAMINE TIME FIELD
C
      TIMTST = BLNK
      CALL MOVINT (CARDIN,47,4)
      CALL MOVCHR (TIMTST)
C
      IF (TIMTST .NE. BLNK) GO TO 150
C
C               WRITE ERROR
C
      WRITE (6,148) (CARDIN(I),I=1,19)
  148 FORMAT (' ERROR -- TIME FIELD BLANK.',/,1X,13A6,A2)
      STOP
C
C               CHECK FOR 'B' INDICATOR
C
  150 RITIM = 4095
      IF (TIMTST .EQ. B81) GO TO 155
C
      DECODE (162,TIMTST) RITIM
  152 FORMAT (I4)
C
C               DO OTHER TIME FIELD
C
  155 TIMTST = BLNK
      CALL MOVINT (CARDIN,58,4)
```

```
      CALL MOVCHR (TIMTST)
C
      IF (TIMTST .NE. BLNK) GO TO 160
C
C               WRITE ERROR
C
      WRITE (6,148) (CARDIN(I),I=1,19)
      STOP
C
C               CHECK FOR '8'
C
  160 RFTIM = 4095
      IF (TIMTST .EQ. B81) GO TO 162
C
      DECODE (152,TIMTST) RFTIM
C
C
C               READ A CARD & PROCESS 20,30,40,50 TYPES
C
  162 CALL CRDRED (CARDIN,SEQ,CRDTYP)
      NCON = 0
      NPAR = 0
      NVAR = 0
      CALL COPAVA (CRDTYP,SEQ,CARDIN)
C
C
C
C
      CALL NTRAN (11,1,LCRUN,PORA,LSTAT)
      CALL NTRAN (11,22)
      IF (LSTAT .GT. 0) GO TO 170
C
C               WRITE ERROR
C
      WRITE (6,165) LSTAT
  165 FORMAT (' ERROR --ARUN WRITE TO 11',I10)
      STOP
C
C               NEXT COMMON 'CPV'
C
  170 CALL NTRAN (11,1,LCCPV,NCON,LSTAT)
      CALL NTRAN (11,22)
      IF (LSTAT .GT. 0 ) GO TO 178
C
C               WRITE ERROR
C
      WRITE (6,175) LSTAT
  175 FORMAT (' ERROR --ACPV WRITE TO 11.',I10)
      STOP
C
C          WRITE DATA OUT
C
  178 CALL NTRAN (11,1,NDATA,DATA(1),LSTAT)
      CALL NTRAN (11,22)
      IF (LSTAT .GT. 0) GO TO 179
C
C
```

```
C          WRITE ERROR
      WRITE (6,181) LSTAT,NDATA
181 FORMAT (' ERROR -- DATA WRITE TO TT',2I10)
      STOP
C
C          GO BACK & READ NEXT CARD
C
  179 GO TO 27
C
C
C          -- END OF PROCESSING FOR ACOUSTIC RUNS
C
C
C
C          REWIND UNIT 11
C
  176 CALL NTRAN (11,10)
C
      SAILAT = -90000
      SAIFLG = 0
      SAIDAT = 991232
      SAITIM = 4095
      SAFLAT = 90000
      SAFLON = 179600
      SAFDAT = 0
      SAFTIM = -4095
      SASTAT(1) = 0
      SASTAT(2) = 0
C
      LSTLTH = 0          @ INIT LAST LENGTH FOR DATA POINTERS
C
C•••••••••••••••••••••• INIT IOTPTR TO PROPER WORD BEFORE DO LOOP ••••••••••.••••
      IOTPTR = 11
      DO 230 I=1,NRUNS
C
C                READ IN DATA FOR A RUN
C
      CALL NTRAN (11,2,LCRUN,PORA,LSTAT)
      CALL NTRAN (11,22)
      IF (LSTAT .GT. 0) GO TO 180
C
      WRITE (6,177) LSTAT,I,NRUNS
  177 FORMAT (' ERROR -- ARUN READ FROM TT',3I10)
      STOP
C
  180 CALL NTRAN (11,2,LCCPV,NCON,LSTAT)
      CALL NTRAN (11,22)
      IF (LSTAT .GT. 0) GO TO 190
C
      WRITE (6,182) LSTAT
  182 FORMAT (' ERROR -- ACPV READ FROM TT',I10)
      STOP
C
  190 CALL NTRAN (11,2,NDATA,DATA(1),LSTAT)
      CALL NTRAN (11,22)
      IF (LSTAT .GT. 0) GO TO 191
C
```

```
C           WRITE ERROR
C
            WRITE (6,192) LSTAT,NDATA
  192 FORMAT (' ERROR -- DATA READ FROM ',2I10)
            STOP
C           CONVERT LAT,LONG,DATE TO PROPER FORM
C
  191 IF (RINS .EQ. 'S') RILAT = -RILAT
            IF (RFNS .EQ. 'S') RFLAT = -RFLAT
            IF (RIEW .EQ. 'W') RILON = -RILON
            IF (RFEW .EQ. 'W') RFLON = -RFLON
C
            RIDAY = RIDAY + RIMO*100 + RIYR*10000
            RFDAY = RFDAY + RFMO*100 + RFYR*10000
C
            CALL ACSTAT (RSTAT)
C
            IF (RILAT .GT. SAILAT) SAILAT = RILAT
            IF (RFLAT .GT. SAILAT) SAILAT = RFLAT
            IF (RILAT .LT. SAFLAT) SAFLAT = RILAT
            IF (RFLAT .LT. SAFLAT) SAFLAT = RFLAT
C
            IF (SAIFLG .NE. 0) GO TO 193
            SAIFLG = 1
            SAILON = EAST (RILON,RFLON)
            SAFLON = WEST (RILON,RFLON)
            GO TO 194
  193 SAILON = EAST(RILON,SAILON)
            SAILON = EAST(RFLON,SAILON)
            SAFLON = WEST(RILON,SAFLON)
            SAFLON = WEST(RFLON,SAFLON)
C
  194 IF (RIDAY .LT. SAIDAT) SAIDAT = RIDAY
            IF (RFDAY .GT. SAFDAT) SAFDAT = RFDAY
C
C
  195 SASTAT(1) = OR(SASTAT(1),RSTAT(1))
            SASTAT(2) = OR(SASTAT(2),RSTAT(2))
C
            CALL MAXMIN
C
            CALL ACMPRS (STABFR,IOTPTR,LNTH)
C
C           CHECK SUBSCRIPT OVERFLOW
C
            IF (IOTPTR .LT. 3000) GO TO 197
            WRITE (6,196) IOTPTR,I,NRUNS
  196 FORMAT (' ERROR -- ACOUSTIC STATION TOO BIG!!',3I10)
            STOP
C
  197 J= IOTPTR -LNTH + 9
            FLD(0,18,STABFR(J)) = FLD(18,18,LSTLTH)
            NDATA1 = NDATA + 1
            TEMP(1) = NSCTRS(NDATA1)          @ FUNCTION NSCTRS COMPUTES THE # SCTRS
            FLD(18,18,STABFR(J)) = FLD(18,18,TEMP(1))
            LSTLTH = TEMP(1) + LSTLTH
C
```

```
      230 CONTINUE
C
      DECODE (232,CARDIN) SAND
  232 FORMAT (T9,I2)
      SANRUN = NRUNS
      CALL ASCMPS (STABFR,3)
C
      IOTPTR =IOTPTR -1
      NSEC = NSCTRS(IOTPTR)
C
      FLD(18,18,STABFR(1)) = NSEC
C
      ISTRT = 11
      SPN = NSEC + NXTSEC
      DO 250 I=1,NRUNS
      FLD(0,18,STABFR(ISTRT+9)) = FLD(0,18,STABFR(ISTRT+9))+ SPN
      ISTRT = FLD(18,18,STABFR(ISTRT))
  250 CONTINUE
C
      CALL DSKRIT (NXTSEC,NSEC,STABFR(1))
      PRINT 252
  252 FORMAT (' ACOUSTIC STATION WRITTEN')
C
      IF (LSTADA .NE. 0) GO TO 280
C
      FLD(18,18,SECADR) = FLD(0,18,LSTADC)
      FLD(18,18,MSEC)   = FLD(18,18,LSTADC)
C
      CALL DSKRED (SECADR,MSEC,STABFR)
C
C               UPDATE DOWN PTR TO ACOUSTIC
C
      FLD(0,18,STABFR(16)) = FLD(18,18,NXTSEC)
      FLD(18,18,STABFR(16))= FLD(18,18,NSEC)
C
C               WRITE IT BACK
C
      CALL DSKRIT (SECADR,MSEC,STABFR)
      PRINT 255
  255 FORMAT (' LINK DOWN FROM CRUISE')
C
      LSTADA = STABFR(16)
C
C
      GO TO 290
C
  280 FLD(18,18,SECADR) = FLD( 0,18,LSTADA)
      FLD(18,18,MSEC)   = FLD(18,18,LSTADA)
C
      CALL DSKRED (SECADR,MSEC,STABFR)
C
      FLD(0,18,STABFR(9)) = FLD(18,18,NXTSEC)
      FLD(18,18,STABFR(9))= FLD(18,18,NSEC)
C
C               WRITE STATION BACK OUT
C
      CALL DSKRIT (SECADR,MSEC,STABFR)
```

```
      PRINT 283
  283 FORMAT (' LINK FROM PREV. STATION')
C
      LSTADA = STABFR(9)
C
C         SET NEXT AVAILABLE ADDRESS
C
  290 NXTSEC = NXTSEC + NSEC
C
C
C         CONTINUE ON TO WRITE OUT RUN DATA
C
      CALL NTRAN (11,10)
C
C         DO LOOP TO READ 11 & WRITE DATA ONLY TO DISK
C
      DO 310 I=1,NRUNS
C
C             READ IN DATA
C
      CALL NTRAN (11,2,LCRUN,PORA,LSTAT)
      CALL NTRAN (11,2,LCCPY,NCON,LSTAT)
      CALL NTRAN(11,22)
      CALL NTRAN (11,2,NDATA,DATA(1),LSTAT)
      CALL NTRAN (11,22)
C
      IF (LSTAT .GT. 0) GO TO 300
C
C         WRITE ERROR
C
      WRITE (6,295) LSTAT
  295 FORMAT (' ERROR -- READING CPY ON RUN ACOUSTIC DATA',I10)
      STOP
C
C         COMPUTE NUMBER OF SECTORS FOR JUST DATA -- SHOULD AGREE W/ PREV. LOOP
C
  300 NDATA1 = NDATA +1
C
      NSEC = NSCTRS(NDATA1)
      FLD(0,18,VALRNG(2,30,50)) = 5
      FLD(18,18,VALRNG(2,30,50)) = NSEC
C
      CALL DSKRIT (NXTSEC,NSEC,VALRNG(2,30,50))
C
C         SET NEXT AVAILABLE ADDRESS
C
      NXTSEC = NXTSEC + NSEC
C
  310 CONTINUE
C
C      ** THIS ENDS THE ACOUSTIC PROCESSING FOR ONE STATION **
C
C
C
C
C
```

```
      C          BEGIN PROCESSING OF ENVIRONMENTAL DATA
      C
      C               SET NO. RUNS TO ZERO
      C
  370 NRUNS = 0
      C
      C           REWIND UNIT 11
      C
          CALL NTRAN (11,10)
          NCON = 0
          PRINT 372
  372 FORMAT (' FINISHED WITH ACOUSTICS')
      C
      C          CHECK FOR ENV. RUN CARD
      C
  380 IF (CRDTYP .EQ. 'E1') GO TO 400
      C
      C          WRITE ERROR
      C
          WRITE (6,390) (CARDIN(I),I=1,14)
  390 FORMAT (' ERROR -- ENVIRONMENTAL RUN CARD EXPECTED.',/,1X,13A6,A2)
          STOP
      C
      C          VERIFY SEQUENCE # '10'
      C
  400 IF (SEQ .EQ. '10') GO TO 415
      C
      C          WRITE ERROR
      C
          WRITE (6,410) (CARDIN(I),I=1,14)
  410 FORMAT (' ERROR -- EXPECTED SEQ 10 CARD.',/,1X,13A6,A2)
          STOP
      C
      C          INCREMENT RUN COUNTER & DECODE DATA (EXCEPT TIME)
      C
  415 NRUNS = NRUNS +1
          NCON = 0
          NDATA = 0
          NDATS = 0
      C
          DECODE (420,CARDIN) PGNA,RILAT,RINS,RFLAT,RFNS,RILON,RIGW,RFLON,
         *                    RFEW,RIMO,RIDAY,RIYR,RIZON,RFMO,RFDAY,RFYR,
         *                    RFZON,NAVCOD,DTASRC,CLAS,DESCR,RUNNO
  420 FORMAT (T14,A1,2,I5,A1),2(I6,A1),2(3I2,4X,A1),2I2,8X,I2,I4,
         *        T11,I3)
      C
      C          EXAMINE TIME FIELD  (THIS SECTION SAME AS ACOUSTIC'S)
      C
          TIMTST = BLNK
          CALL MOVINT (CARDIN,47,4)
          CALL MOVCHR (TIMTST)
      C
          IF (TIMTST .NE. BLNK) GO TO 440
      C
      C          WRITE ERROR
      C
          WRITE (6,148) (CARDIN(I),I=1,14)
```

```
      STOP
C
C          CHECK FOR 'B' INDICATOR
C
  440 RITIM = 4095
      IF (TIMTST .EQ. BBI) GO TO 460
C
      DECODE (152,TIMTST) RITIM
C
C          DO OTHER TIME FIELD
C
  460 TIMTST = BLNK
      CALL MOVINT (CARDIN,58,4)
      CALL MOVCHR (TIMTST)
C
      IF (TIMTST .NE. BLNK) GO TO 480
C
C          WRITE ERROR
C
      WRITE (6,148) (CARDIN(I),I=1,14)
      STOP
C
C          CHECK FOR 'B'
C
  480 RFTIM = 4095
      IF (TIMTST .EQ. BBI) GO TO 500
C
      DECODE (152,TIMTST) RFTIM
C
C
C          READ NEXT CARD
C
  500 CALL CRDRED (CARDIN,SEQ,CRDTYP)
C
C          PROCESS '11' TYPE CARD
C
      IF (SEQ .EQ. '11') GO TO 520
C
C          WRITE ERROR
C
      WRITE (6,510) (CARDIN(I),I=1,14)
  510 FORMAT (' ERROR -- EXPECTED SEQ 11 CARD.',/,1X,13A6,A2)
      STOP
C
C          CONTINUE TO DECODE CARD
C
  520 DECODE (525,CARDIN) #AVHT,HTUNIT,#HTMTD,#AVPRD,PRDUNT,PRDMTD,
     *                    #NDDIR,DIRUNT,DIRMTD,#NDVEL,VELUNT,VELMTD,
     *                    $WLHT,SHTUNT,SHTMTD,$WLPRD,SPDUNT,SPDMTD
  525 FORMAT (T16,2(F3.0,2I3),1X,F3.0,2I3,F4.0,2I3,2X,
     *        2(F3.0,2I3))
C
C          CHECK POSSIBLE BLANK FIELDS
C
      IF (HTUNIT .EQ. 0 .AND. #AVHT  .NE. 0.) GO TO 530
      IF (PRDUNT .EQ. 0 .AND. #AVPRD .NE. 0.) GO TO 530
      IF (VELUNT .EQ. 0 .AND. #NDVEL .NE. 0.) GO TO 530
```

```
      IF (SHTUNT .EQ. 0 .AND. SWLHT  .NE. 0.) GO TO 530
      IF (SPDUNT .EQ. 0 .AND. SWLPRD .NE. 0.) GO TO 530
      GO TO 540
C
C          WRITE ERROR
C
  530 WRITE (6,535) (CARDIN(I),I=1,19)
  535 FORMAT (' ERROR -- UNIT FIELD = 0, DATA FIELD NOT',/,1X,13A6,A2)
      STOP
C
C          CLEAR STATUS WORDS
C
  540 NSTAT(1) = 0
      NSTAT(2) = 0
C
C          CONVERT UNITS & MOVE DATA TO CONSTANTS AREA
C
C               #AVE HEIGHT
C
      IF (HTUNIT .EQ. 0) GO TO 550
      CALL STOR(102,#AVHT,HTUNIT,#HTMTD)
C
C               #AVE PERIOD
C
  550 IF (PRDUNT .EQ. 0) GO TO 552
      CALL STOR (103,#AVPRD,PRDUNT,PRDMTD)
C
C               #IND DIRECTION
C
  552 IF(DIRUNT.EQ.0) GO TO 555
      CALL STOR(105,#NDDIR,DIRUNT,DIRMTD)
C
C               #IND VELOCITY
C
  555 IF (VELUNT .EQ. 0) GO TO 560
      CALL STOR (106,#NDVEL,VELUNT,VELMTD)
C
C               SWELL HEIGHT
C
  560 IF (SHTUNT .EQ. 0) GO TO 565
      CALL STOR (108,SWLHT,SHTUNT,SHTMTD)
C
C               SWELL PERIOD
C
  565 IF (SPDUNT .EQ. 0) GO TO 600
      CALL STOR (109,SWLPRD,SPDUNT,SPDMTD)
C
C          PROCESS #AVE DIRECTION FIELD
C
  600 CALL MOVINT (CARDIN,19,2)
      TEMP(1) = BLNK
      CALL MOVCHR (TEMP)
      IF (TEMP(1) .EQ. BB2) GO TO 620
      IF (TEMP(1) .NE. BLNK)GO TO 610
      WRITE (6,605) (CARDIN(I),I=1,19)
  605 FORMAT (' ERROR -- #AVE DIRECTION FIELD BLANK',/,1X,13A6,A2)
      STOP
```

```
      617 DECODE (611,TEMP) #AVDIR
      611 FORMAT(F2.0)
          CALL STOR (101 ,#AVDIR,1,0)
C
C           PROCESS SEA STATE FIELD
C
      620 CALL MOVINT (CARDIN,34,1)
          TEMP(1) = BLNK
          CALL MOVCHR (TEMP)
          IF (TEMP(1) .EQ. BB3) GO TO 680
          IF (TEMP(1) .NE. BLNK)GO TO 630
          #RITE (6,625) (CARDIN(I),I=1,14)
      625 FORMAT(' ERROR -- SEA STATE FIELD BLANK.',/,1X,13A6,A2)
          STOP
      630 DECODE (631,TEMP) SEASTA
      631 FORMAT (F1.0)
          CALL STOR (104,SEASTA,1,0)
C
C           PROCESS S#ELL DIRECTION FIELD
C
      680 CALL MOVINT(CARDIN,54,2)
          CALL MOVCHR (TEMP)
          IF (TEMP(1) .EQ. BB2) GO TO 700
          IF (TEMP(1) .NE. BLNK)GO TO 690
          #RITE (6,685) (CARDIN(I),I=1,14)
      685 FORMAT (' ERROR -- S#ELL DIRECTION FIELD BLANK.',/,1X,13A6,A2)
          STOP
      690 DECODE (611,TEMP) S#LDIR
          CALL STOR (107,S#LDIR,1,0)
C
C           PROCESS #EATHER CODE FIELD
C
      700 CALL MOVINT (CARDIN,74,2)
          CALL MOVCHR (TEMP)
          IF (TEMP(1) .EQ. BB2) GO TO 720
          IF (TEMP(1) .NE. BLNK)GO TO 710
          #RITE (6,705) (CARDIN(I),I=1,14)
      705 FORMAT (' ERROR -- #EATHER CODE FIELD BLANK.',/,1X,13A6,A2)
          STOP
      710 DECODE (611,TEMP) #EATHR
          CALL STOR (110,#EATHR,1,0)
C
C           PROCESS BOTTOM SLOPE FIELD
C
      720 CALL MOVINT (CARDIN,76,3)
          TEMP(1) = BLNK
          CALL MOVCHR (TEMP)
          BTMSLP = 0.
          IF (TEMP(1) .EQ. BB4) GO TO 743
          IF (TEMP(1) .NE. BLNK)GO TO 730
          #RITE (6,725) (CARDIN(I),I=1,14)
      725 FORMAT (' ERROR -- BOTTOM SLOPE FIELD BLANK.',/,1X,13A6,A2)
          STOP
      730 CALL MOVINT (CARDIN,76,1)
          TEMP(1) = BLNK
          CALL MOVCHR (TEMP)
          IF (TEMP(1) .NE. BLNK) BTMSLP = 1.
```

```
      CALL MOVCHR (TEMP)
      IF (TEMP(1) .EQ. BLNK) GO TO 735
      IF (BTMSLP .EQ. 0.) GO TO 733
      WRITE (6,731) (CARDIN(I),I=1,14)
  731 FORMAT (' ERROR -- MULTIPLE CHECKS IN BOTTOM SLOPE FIELD.',/,1X,
     .        13A6,A2)
      STOP
  733 BTMSLP = 2.
  735 CALL MOVCHR (TEMP)
      IF (TEMP(1) .EQ. BLNK) GO TO 740
      IF (BTMSLP .EQ. 0.) GO TO 737
      WRITE (6,731) (CARDIN(I),I=1,14)
      STOP
  737 BTMSLP = 3.
  740 CALL STOR (111,BTMSLP,1,0)
C
C         PROCESS BATHEMETRY FIELD
C
  743 CALL MOVINT (CARDIN,79,1)
      TEMP(1) = BLNK
      CALL MOVCHR (TEMP)
      IF (TEMP(1) .NE. 'Y') GO TO 741
      RSTAT(2) = RSTAT(2) + 4
      GO TO 750
  741 IF (TEMP(1) .EQ. 'N') GO TO 750
      WRITE (6,742) (CARDIN(I),I=1,14)
  742 FORMAT (' ERROR -- BATHEMETRY FIELD NOT  Y  OR  N',/,1X,13A6,A2)
      STOP
C
C       -- END OF PROCESSING ERII CARD --
C
C
C         READ NEXT CARD
C
  750 CALL CRDRED (CARDIN,SEQ,CRDTYP)
C
C         VERIFY  SEQ = '12'
C
      IF (SEQ .EQ. '12') GO TO 760
C
C         WRITE ERROR
C
      WRITE (6,755) (CARDIN(I),I=1,14)
  755 FORMAT (' ERROR -- EXPECTED SEQUENCE 12 CARD.',/,1X,13A6,A2)
      STOP
  760 DECODE(761,CARDIN) BOTDPH,DPHUNT,DPHMTD
  761 FORMAT(T14,F7.0,213)
      IF(DPHUNT.EQ.0.AND.BOTDPH.NE.0) GO TO 530
      IF(DPHUNT.EQ.0) GO TO 7615
      CALL STOR(113,BOTDPH,DPHUNT,DPHMTD)
C
C         MOVE DATA FIELD
C
 7615 CALL MOVINT (CARDIN,27,54)
      CALL MOVCHR (INFO)
C
C         CLEAR END OF FILE FLAG
```

```
C
      EOF = 0
C
C          READ NEXT CARD & CHECK END OF FILE
C
      READ(9,762,END=765) (CARDIN(I),I=1,14)
  762 FORMAT (13A6,A2)
      IF (FLD(0,6,CARDIN(1)) .EQ. FLD(0,6,'@')) GO TO 765
      FLD(0,12,SEQ) = FLD(6,12,CARDIN(1))
      GO TO 770
C
C          SET EOF FLAG & CLEAR SEQ
C
  765 EOF = 1
      SEQ = '00'
C
  770 NPAR = 0
      NVAR = 0
C
C          CHECK NEW CARD FOR 20, 30, 40 TYPE
C
      IF (.NOT. (SEQ .EQ. '20' .OR. SEQ .EQ. '30' .OR. SEQ .EQ. '40'))
     .          GO TO 775
C
C          PROCESS 'CPV' TYPE CARDS
C
      FLD(0,12,CRDTYP) = FLD(0,12,CARDIN(1))
C
      CALL COPAVA (CRDTYP,SEQ,CARDIN)
C
C          READ NEXT CARD & CHECK FOR EOF
C
      READ (9,762,END=780) (CARDIN(I),I=1,14)
      IF (FLD(0,6,CARDIN(1)) .EQ. FLD(0,6,'@')) GO TO 780
      FLD(0,12,CRDTYP) = FLD(0,12,CARDIN(1))
      FLD(0,12,SEQ)    = FLD(6,12,CARDIN(1))
      GO TO 800
C
  775 '  (NCON .EQ. 0) GO TO 800
      DO 778 I=1,NCON
      CALL CONVRT(CON(3,I),CON(1,I),CON(2,I),UNITS)
      IF (UNITS(1) .EQ. 'UNITS ') WRITE(6,777) (CON(J,1),J=1,3)
  777 FORMAT (' CONSTANTS ',2I5,F21.7)
  778 CONTINUE
      GO TO 800
C
C               SET EOF FLAG
C
  780 EOF = 1
C
C
C          DO COMMON 'RUN'
C
  800 CALL NTRAN (11,1,LCRUN,PORA,LSTAT)
      CALL NTRAN (11,22)
      IF (LSTAT .GT. 0) GO TO 810
C
```

```
      C                 WRITE ERROR
      C
            WRITE (6,805) LSTAT
  805 FORMAT (' ERROR -- ERUN WRITE TO II',I10)
            STOP
      C
      C                 NEXT COMMON CPV
      C
  810 CALL NTRAN (11,1,LCCPV,NCON,LSTAT)
            CALL NTRAN (11,22)
            IF (LSTAT .GT. 0) GO TO 820
      C
      C                 WRITE ERROR
      C
            WRITE (6,815) LSTAT
  815 FORMAT (' ERROR -- ECPV WRITE TO II',I10)
            STOP
      C
      C           WRITE OUT DATA
      C
  820 CALL NTRAN (11,1,NDATA,DATA(1),LSTAT)
            CALL NTRAN (11,22)
            IF (LSTAT .GE. 0) GO TO 821
      C
      C           WRITE ERROR
      C
            WRITE (6,825) LSTAT
  825 FORMAT (' ERROR -- EDATA WRITE TO II',I10)
            STOP
      C
      C           CHECK EOF FLAG. IF NOT SET, GO PROCESS CARD JUST READ
      C
  821 IF (EOF .EQ. 0) GO TO 380
      C
      C
      C        -- END OF PROCESSING FOR ENVIRONMENTAL RUNS
      C
      C
      C           REWIND UNIT 11
      C
            CALL NTRAN (11,10)
      C
      C           INIT FOR STATION SUMMARY
      C
            SAILAT = -90000
            SAIFLG = 0
            SAIDAT = 991232
            SAITIM = 4095
            SAFLAT = 90000
            SAFLON = 179600
            SAFDAT = 0
            SAFTIM = -4095
            SASTAT(1) = 0
            SASTAT(2) = 0
      C
            LSTLTH =0           @ INIT LAST LENGTH FOR RUN DATA POINTERS
      C
```

```
C          --DO LOOP TO READ IN ENVIRONMENTAL RUNS
C
           IOTPTR = 1
           DO 1000 I=1,NRUNS
C
C              READ IN DATA FOR A RUN
C
           CALL NTRAN (11,2,LCRUN,PORA,LSTAT)
           CALL NTRAN (11,22)
           IF (LSTAT .GT. 0) GO TO 880
C
           WRITE (6,877) LSTAT
       877 FORMAT (' ERROR -- ERUN READ FROM 11',I10)
           STOP
C
       880 CALL NTRAN (11,2,LCCPV,NCON,LSTAT)
           CALL NTRAN (11,22)
           IF (LSTAT .GT. 0) GO TO 890
C
           WRITE (6,882) LSTAT
       882 FORMAT (' ERROR -- ECPV READ FROM 11',I10)
           STOP
C
       890 CALL NTRAN (11,2,NDATA,DATA(1),LSTAT)
           CALL NTRAN (11,22)
           IF (LSTAT .GE. 0) GO TO 891
C
C
C              WRITE ERROR
C
           WRITE (6,892) LSTAT
       892 FORMAT (' ERROR -- EDATA READ FROM 11',I10)
           STOP
C
C              CONVERT LAT, LONG, DATE TO PROPER FORM
C
       891 IF (RINS .EQ. 'S') RILAT = -RILAT
           IF (RFNS .EQ. 'S') RFLAT = -RFLAT
           IF (RIEW .EQ. 'W') RILON = -RILON
           IF (RFEW .EQ. 'W') RFLON = -RFLON
           RIDAY = RIDAY + RIMO*100 + RIYR*10000
           RFDAY = RFDAY + RFMO*100 + RFYR*10000
C
           CALL ENSTAT (RSTAT)
C
C              UPDATE STATION SUMMARY
C
           IF (RILAT .GT. SAILAT) SAILAT = RILAT
           IF (RFLAT .GT. SAILAT) SAILAT = RFLAT
           IF (RILAT .LT. SAFLAT) SAFLAT = RILAT
           IF (RFLAT .LT. SAFLAT) SAFLAT = RFLAT
C
           IF (SAIFLG .NE. 0) GO TO 893
           SAIFLG = 1
           SAILON = EAST(RILON,RFLON)
           SAFLON = WEST(RILON,RFLON)
           GO TO 894
```

```
      893 SAILON = EAST(RILON,SAILON)
          SAILON = EAST(RFLON,SAILON)
          SAFLON = WEST(RILON,SAFLON)
          SAFLON = WEST(RFLON,SAFLON)
C
      894 IF (RIDAY .LT. SAIDAT) SAIDAT = RIDAY
          IF (RFDAY .GT. SAFDAT) SAFDAT = RFDAY
C
C
      895 SASTAT(1) = OR(SASTAT(1),RSTAT(1))
          SASTAT(2) = OR(SASTAT(2),RSTAT(2))
C
          CALL MAXMIN
C
C         CALL ROUTINE TO COMPRESS DATA INTO OUTPUT ARRAY
C
          CALL ACMPRS (STABFR,IOTPTR,LNTH)
C
C           CHECK SUBSCRIPT OVERFLOW
C
          IF (IOTPTR .LT. 3000) GO TO 897
          WRITE (6,896) IOTPTR,I,NRUNS
      896 FORMAT(' ERROR -- ENV STATION TOO BIG!!',3I10)
          STOP
C
      897 J = IOTPTR - LNTH + 9
          FLD(0,18,STABFR(J)) = FLD(18,18,LSTLTH)
          NDATA1 = NDATA + 10  @ NDATA + 1 + NO. WORDS IN INFO FILED..
          TEMP(1) = NSCTRS(NDATA1)
          FLD(18,18,STABFR(J)) = FLD(18,18,TEMP(1))
          LSTLTH = TEMP(1) +LSTLTH
C
 1000 CONTINUE
C
          SAVRUN = NRUNS
          CALL ASCMPS (STABFR,9)
C
          IOTPTR = IOTPTR -1
          NSEC = NSCTRS (IOTPTR)
C
          FLD(18,18,STABFR(1)) = NSEC
C
          ISTRT = 1
          SPN = NSEC + NXTSEC
          DO 1050 I=1,NRUNS
          FLD(0,18,STABFR(ISTRT+9)) = FLD(0,18,STABFR(ISTRT+9)) + SPN
          ISTRT = FLD(18,18,STABFR(ISTRT))
 1050 CONTINUE
C
          PRINT 1053
 1053 FORMAT (' WRITE ENV STATION')
          CALL DSKRIT (NXTSEC,NSEC,STABFR)
C
          IF (LSTADR .NE. 0) GO TO 1080
C
          FLD(18,18,SECADR) = FLD(00,18,LSTADC)
          FLD(18,18,NSEC)   = FLD(18,18,LSTADC)
```

```
1138.      C
1139.            CALL DSKRED (SECADR,MSEC,STABFR)
1140.      C
1141.      C           UPDATE DOWN ENV. POINTER
1142.      C
1143.            FLD(0,18,STABFR(17)) = FLD(18,18,NXTSEC)
1144.            FLD(18,18,STABFR(17))= FLD(18,18,NSEC)
1145.      C
1146.      C           WRITE IT BACK
1147.      C
1148.            CALL DSKRIT (SECADR,MSEC,STABFR)
1149.            PRINT 1057
1150.       1057 FORMAT (' LINK DOWN ENV')
1151.      C
1152.            LSTADE = STABFR(17)
1153.      C
1154.      C
1155.            GO TO 1090
1156.      C
1157.       1080 FLD(18,18,SECADR) = FLD( 0,18,LSTADE)
1158.            FLD(18,18,MSEC)   = FLD(18,18,LSTADE)
1159.      C
1160.            CALL DSKRED (SECADR,MSEC,STABFR)
1161.      C
1162.      C           UPDATE PTR TO NEXT STATION
1163.      C
1164.            FLD( 0,18,STABFR(9)) = FLD(18,18,NXTSEC)
1165.            FLD(18,18,STABFR(9)) = FLD(18,18,NSEC)
1166.      C
1167.      C           WRITE ENV STATION BACK OUT
1168.      C
1169.            CALL DSKRIT (SECADR,MSEC,STABFR)
1170.            PRINT 1083
1171.       1083 FORMAT (' LINK FROM ENV')
1172.      C
1173.            LSTADE = STABFR(9)
1174.      C
1175.      C           SET NEXT AVAILABLE ADDRESS
1176.      C
1177.       1090 NXTSEC = NXTSEC + NSEC
1178.      C
1179.      C
1180.      C           CONTINUE ON TO WRITE RUN DATA TO DISK
1181.      C
1182.            CALL NTRAN (11,10)
1183.      C
1184.      C           DO LOOP TO READ 11 & WRITE SHIP NAME DATA & VARIABLE DATA TO
1185.      C           DISK
1186.      C
1187.            DO 1110 I=1,NRUNS
1188.      C
1189.      C           READ IN DATA
1190.      C
1191.            CALL NTRAN (11,2,LCRUN,PORA,LSTAT)
1192.            CALL NTRAN (11,2,LCCPV,NCON,LSTAT)
1193.            CALL NTRAN (11,22)
1194.            CALL NTRAN (11,2,NDATA,DATA(1),LSTAT)
```

```
      CALL NTRAN (11,22)
C
C
      IF (LSTAT .GE. 0) GO TO 1100
C
C          WRITE ERROR
C
      WRITE (6,1095) LSTAT
 1095 FORMAT (' ERROR -- READING CPV OR RUN ENV DATA',I3)
      STOP
C
C          MOVE SHIP NAME DATA TO LAST 9 WORDS IN 'CPV' BEFORE 'DATA'.
C
 1100 DO 1105 J=1,9
      IEQV(2991+J) = INFO(J)
 1105 CONTINUE
C
C          COMPUTE NO. OF SECTORS TO WRITE OUT
C
      TEMP(1) = NDATA + 10
      NSEC = NSCTRS(TEMP(1))
C
      FLD( 0,18,IEQV(2991)) = 6
      FLD(18,18,IEQV(2991)) = NSEC
C
C
      CALL DSKRIT (NXTSEC,NSEC,IEQV(2991))
C
C
      NXTSEC = NXTSEC + NSEC
C
 1110 CONTINUE
C
C
C
C
      FLD(18,18,SECADR) = FLD( 0,18,LSTADA)
      FLD(18,18,MSEC)   = FLD(18,18,LSTADA)
C
C               READ IN ACOUSTIC STATION
C
      CALL DSKRED (SECADR,MSEC,STABFR)
C
      STABFR(10) = LSTADE
C
C               WRITE ACOUSTIC STATION BACK
C
      CALL DSKRIT (SECADR,MSEC,STABFR)
C
      FLD(18,18,SECADR) = FLD( 0,18,LSTADE)
      FLD(18,18,MSEC)   = FLD(18,18,LSTADE)
C
C               READ IN ENVIRONMENTAL STATION
C
      CALL DSKRED (SECADR,MSEC,STABFR)
C
      STABFR(10) = LSTADA
```

```
C
C                WRITE ENV STATION BACK
C
      CALL DSKRIT (SECADR,NSEC,STABFR)
C
      CALL CRUSUM (DATA,STABFR)
      PRINT 1150
 1150 FORMAT (' PAST CRUSUM')
C
      CALL EXPSUM (DATA,STABFR)
      PRINT 1159
 1159 FORMAT (' PAST EXPSUM')
C
C          DECREMENT STATION COUNTER BY 1
C
      NOSTA = NOSTA -1
C
C          IF THAT WAS LAST STATION DECREMENT CRUISE COUNTER
C
      IF (NOSTA .NE. 0) GO TO 1175
      NOCRU = NOCRU -1
      LSTADA = 0
      LSTADE = 0
C
C          UPDATE HEADER & WRITE IT BACK
C
 1175 IDATE = NDATE
      CALL DSKRIT (0,1,SECTZ)
C
C          DECREMENT NO. OF FILES PROCESSED
C
      NOFILP = NOFILP -1
C
C          IF NOT FINISHED, DO NEXT FILE
C
      WRITE (6,1200)
 1200 FORMAT (' *** ANOTHER STATION COMPLETED ***')
C
      IF (NOFILP .NE. 0) GO TO 5
C
      STOP
      END
```

```
@ELT,L      N.MAXMIN,.MAXMIN

      SUBROUTINE MAXMIN
C
C         MAXMIN (MAXIMUM-MINIMUM) COMPUTES THE MAX & MIN VALUES FOR
C         EACH VARIABLE IN EACH DATA SET IN 'DATA' OF COMMON /CPV/.
C         THE MAX MIN VALUES ARE STORED IN 'VALRNG' IN COMMON /CPV/.
C
C              THIS PROGRAM WAS CHALKBOARD 'DEBUGGED' BY DENNIS BOWEN
C              AND JEFF ANDERSON ON 22 MAY 1974.
C                                                        GFA
      IMPLICIT INTEGER (A-Z)
C
C
      COMMON /CPV    /NCON,CON(4,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
     *                NDATS,NROWS(50),NDATA,VALRNG(2,30,50),DATA(30000)
      DIMENSION ICON(4,30),IPAR(30,30),IVLRNG(2,30,50)
      EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
     *            (VALRNG(1,1,1),IVLRNG(1,1,1))
      REAL CON,PAR,DATA,VALRNG
C
C
      REAL AMAX,AMIN
      DATA ANULL /07777777777776/
C
      L = 1
C
      DO 30 J=1,NDATS
      N = NROWS(J) * NVAR + L-1
      M = NVAR +L -1
C
          DO 20 K=L,M
          KK = K-L+1
          AMIN = 10E37
          AMAX = -10E37
C
              DO 10 I=K,N,NVAR
      IF (DATA(I) .EQ. ANULL) GO TO 10
              IF (DATA(I) .LT. AMIN) AMIN = DATA(I)
              IF (DATA(I) .GT. AMAX) AMAX = DATA(I)
   10         CONTINUE
C
          VALRNG(1,KK,J) = AMIN
          VALRNG(2,KK,J) = AMAX
   20     CONTINUE
C
      L = N+1
   30 CONTINUE
C
C
      RETURN
      END
```

```
RELT,L      N.MOVINT,.MOVINT

      SUBROUTINE MOVINT(FROM,ISTRT,NOCHR)
C
C         MOVINT (MOVE INITIALIZE) INITIALIZES VARIABLES FOR FUTURE
C         CALLS TO MOVCHR.
C
C               FROM   - ADDRESS OF FIELD CHARACTERS ARE TO BE MOVED
C                        FROM (STORED 6 CHAR. PER WORD)
C               ISTRT  - CHARACTER COUNT WHERE MOVE IS TO START FROM
C               NOCHR  - NUMBER OF CHARACTERS TO MOVE EACH TIME
C                        MOVCHR IS CALLED.
C
      DIMENSION FROM(1),TO(1)
C
      IST = ISTRT
      RETURN
C
C
C
      ENTRY MOVCHR (TO)
C
C         MOVCHR (MOVE CHARACTER) COPIES 'NOCHR' NUMBER OF CHARACTERS
C                STARTING WITH 'ISTRT' CHAR IN FROM TO THE FIRST 'NOCHR'
C                CHARACTERS IN 'TO'.
C           EACH TIME MOVCHR IS CALLED AFTER MOVINT HAS BEEN CALLED, THE
C           NEXT 'NOCHR'S ARE COPIED.  MOVCHR KEEPS PROCESSING THE NEXT
C           'NOCHR'S UNTIL RE-INITIALIZED BY A CALL TO MOVINT.
C
C
C
      DO 30 J=1,NOCHR
C
      ISUB = IST / 6 +1
      ICHR = MOD(IST,6)
      IF (ICHR .EQ. 0) ISUB = ISUB -1
      IF (ICHR .EQ. 0) ICHR = 6
      IBT = ICHR * 6 - 6
C
      ISUB1 = J /6 + 1
      IBT1  = MOD(J,6)
      IF (IBT1 .EQ. 0) ISUB1 = ISUB1 -1
      IF (IBT1 .EQ. 0) IBT1 = 6
      IBT1 = IBT1 * 6 - 6
C
      FLD (IBT1,6,TO(ISUB1)) = FLD(IBT,6,FROM(ISUB))
      IST = IST + 1
C
   30 CONTINUE
C
      RETURN
      END
```

```
@ELT,L      N.NSCTRS,,NSCTRS

      1.          FUNCTION NSCTRS (IARG)
      2.      C
      3.          ITMP = IARG / 28 +1
      4.          IF (MOD(28,IARG) .EQ. 0) ITMP = ITMP-1
      5.          NSCTRS = ITMP
      6.          RETURN
      7.          END
```

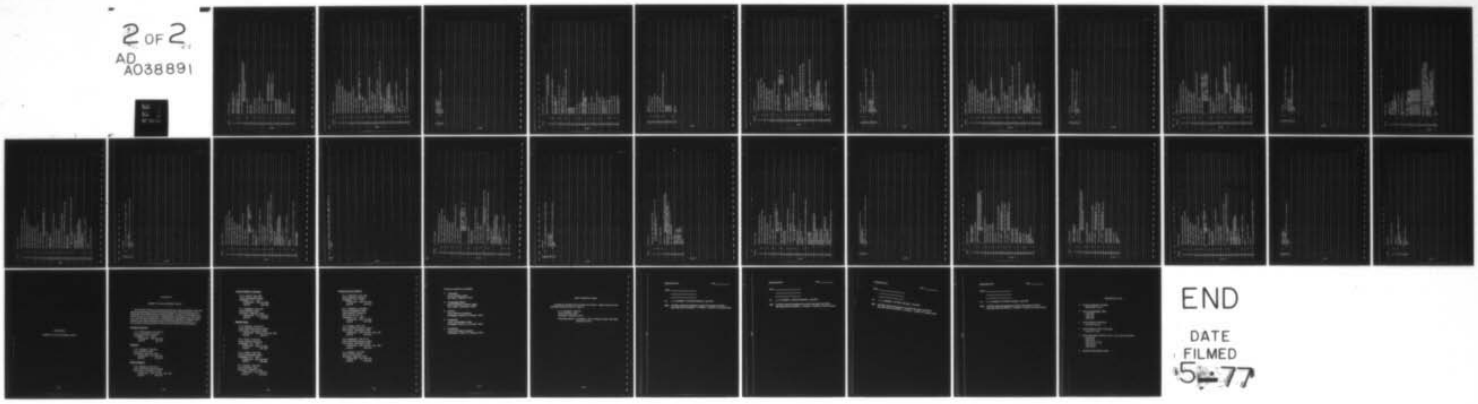
```
@ELT,L      N,PCKCRU,,PCKCRU

      1.            SUBROUTINE PCKCRU(DATA,NSEC)
      2.            IMPLICIT INTEGER(A-Z)
      3.      C
      4.            DIMENSION DATA(1)
      5.      C
      6.      C     DATA IS PACKED FROM CRUISE COMMON.
      7.      C
      8.      C
      9.            COMMON /CRUISE/CRUNO,PNXTC,CAILAT,CAILON,CAIDAT,CAFLAT,CAFLON,
     10.           *               CAFDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,
     11.           *               CASTAT(2),CESTAT(2),PACS,PENS,NCARDS,NOTES(14,132),
     12.           *               CAFLG
     13.      C
     14.            EQUIVALENCE (INOTES(1),NOTES(1,1))
     15.            DIMENSION INOTES(1)
     16.      C
     17.      C     COMPUTE NUMBER OF WORDS IN NOTES AND CLEAN OUT ARRAY 'DATA'.
     18.      C
     19.            N=14*NCARDS+17
     20.            DO 1 I=1,N
     21.          1 DATA(I)=0
     22.      C
     23.            FLD(16,1,DATA(1))=1
     24.      C
     25.      C     COMPUTE AND PACK NUMBER OF SECTORS TO BE USED.
     26.      C
     27.            NSEC=NSCTRS(N)
     28.            FLD(18,18,DATA(1))=NSEC
     29.      C
     30.      C     PACK CRUISE NUMBER.
     31.      C
     32.            FLD(0,18,DATA(2)) = CRUNO
     33.      C
     34.      C          PACK NUMBER OF CARDS.
     35.      C
     36.            FLD (18,18,DATA(2)) = NCARDS
     37.      C
     38.      C     PACK POINTER TO NEXT CRUISE.
     39.      C
     40.            DATA(3)=PNXTC
     41.      C
     42.            CALL PKAREA(' ',CAILAT,CAILON,CAIDAT,0,' ',
     43.           *               CAFLAT,CAFLON,CAFDAT,0,' ',DATA(4))
     44.      C
     45.      C     PACK ENVIRONMENT INITIAL AND FINAL POINT BOUNDS.
     46.      C
     47.            CALL PKAREA(' ',CEILAT,CEILON,CEIDAT,0,' ',
     48.           *               CEFLAT,CEFLON,CEFDAT,0,' ',DATA(8))
     49.      C
     50.            DATA(12)=CASTAT(1)
     51.            DATA(13)=CASTAT(2)
     52.      C
     53.            DATA(14)=CESTAT(1)
     54.            DATA(15)=CESTAT(2)
```

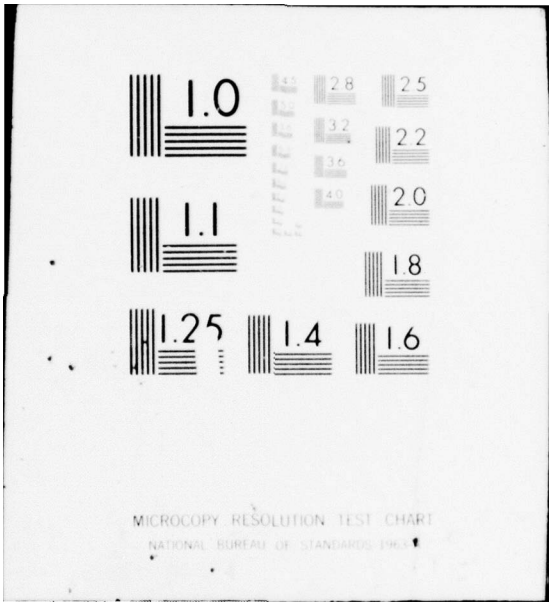
```
      C
            DATA(16)=PACS
            DATA(17)=PENS
      C
      C     PACK NOTES.
      C
            N=14*NCARDS
            DO 2 I=1,N
          2 DATA(I+17)=INOTES(I)  BMAY NEED EQUIV(INOTES(1),NOTES(1,1))
            RETURN
            END
```

AD-A038 891 NAVAL SEA SYSTEMS COMMAND WASHINGTON D C F/G 17/1
NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK - NAVDAB - IN SUP--ETC(U)
DEC 75
UNCLASSIFIED NAVSEA-06H1/036-EVA/MOST- NL

2 OF 2
AD A038891



END
DATE FILMED
5-77


1.0
1.1
1.25
1.4
1.6
2.8
2.5
3.2
2.2
3.6
2.0
4.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963

```
@ELT,L      N.PCKEXP,,PCKEXP

      1.            SUBROUTINE PCKEXP(DATA,NSEC)
      2.            IMPLICIT INTEGER(A-Z)
      3.      C
      4.      C     DATA IS PACKED INTO 'DATA'.
      5.            DIMENSION DATA(1)
      6.      C     DATA IS PACKED FROM EXPERIMENT COMMON.
      7.      C
      8.            COMMON /EXPMNT/EXNO,PNXTE,PCRU,EAILAT,EAILON,EAIDAT,EAFLAT,EAFLON,
      9.           *               EAFDAT,EEILAT,EEILON,EEIDAT,EEFLAT,EEFLON,EEFDAT,
     10.           *               EASTAT(2),EESTAT(2),EAFLG
     11.      C
     12.      C     CLEAN ARRAY 'DATA'.
     13.      C
     14.            DO 1 I=1,20
     15.          1 DATA(I)=0
     16.      C
     17.      C     SET LEVEL TYPE BIT.
     18.      C
     19.            FLD(17,1,DATA(1))=1
     20.      C
     21.      C     SET NUMBER OF SECTORS FIELD.
     22.      C
     23.            FLD(35,1,DATA(1))=1
     24.      C
     25.            DATA(2)=EXNO
     26.      C
     27.            CALL PKAREA(' ',EAILAT,EAILON,EAIDAT,0,' ',
     28.           *               EAFLAT,EAFLON,EAFDAT,0,' ',DATA(3))
     29.      C
     30.            CALL PKAREA(' ',EEILAT,EEILON,EEIDAT,0,' ',
     31.           *               EEFLAT,EEFLON,EEFDAT,0,' ',DATA(7))
     32.      C
     33.            DATA(11)=EASTAT(1)
     34.            DATA(12)=EASTAT(2)
     35.      C
     36.            DATA(13)=EESTAT(1)
     37.            DATA(14)=EESTAT(2)
     38.      C
     39.            DATA(15)=PNXTE
     40.      C
     41.            DATA(16)=PCRU
     42.      C
     43.      C     ONLY USED ONE SECTOR, SO...
     44.      C
     45.            NSEC=1
     46.            RETURN
     47.            END
```

```
@FOR,SA     N.PERCNT,.PERCNT

      SUBROUTINE PERCNT(VALUE,UNICDE,UNITS)
C
C     TABLE OF PERCNT MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,3)/42,43,44/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,3)/'PER CE','NT    ',
     .                '(0/00)','      ','PPM   ','      '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='PERCEN'
      UNITS(4)='T     '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-42
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,3)/1.000,0.100,1.00-4/
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,3
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
    2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
```

```
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN PER CENT TABLE.')
      RETURN
      END
```

```
@ELT,L      N.PKAREA,.PKAREA

      1.            SUBROUTINE PKAREA(PORA,ILAT,ILON,IDAT,ITIM,IZON,
      2.           *                  FLAT,FLON,FDAT,FTIM,FZON,IOT)
      3.      C
      4.      C
      5.      C     MAKE ALL VARIABLES INTEGER AND DIMENSION IOT.
      6.      C
      7.            IMPLICIT INTEGER (A-Z)
      8.            DIMENSION IOT(4)
      9.      C
     10.      C     SET POINT OR AREA BIT ('ON' FOR AREA)
     11.      C
     12.            FLD(0,1,IOT(1)) = 0
     13.            IF(PORA.EQ.'A') FLD(0,1,IOT(1))=1
     14.      C
     15.      C     INITALIZE INTERMEDIATE VARIABLES TO INITIAL POINT VALUES.
     16.      C
     17.            IT1=IDAT
     18.            IT2=ITIM
     19.            IT3=IZON
     20.            IT4=ILAT
     21.            IT5=ILON
     22.      C
     23.            DO 4 I=1,3,2
     24.      C
     25.      C     PACK THE DATE.
     26.      C
     27.            ITMP=IT1/10000
     28.            ITMP=(IT1-ITMP*10000)*100+ITMP
     29.            IF(IT1.EQ.0) ITMP=6
     30.            FLD(1,17,IOT(I))=ITMP
     31.      C
     32.      C     PACK THE TIME.
     33.      C
     34.            FLD(18,12,IOT(I))=IT2
     35.      C
     36.      C     PACK THE ZONE.
     37.      C
     38.            FLD(30,5,IOT(I))=FLD(1,5,IT3)
     39.      C
     40.      C     SET LATITUDE SIGN BIT
     41.      C
     42.            FLD(35,1,IOT(I)) = 0
     43.            IF(IT4.GE.0) GOTO 2
     44.            IT4=-IT4
     45.            FLD(35,1,IOT(I))=1
     46.      C
     47.      C     PACK THE LATITUDE.
     48.      C
     49.          2 FLD(0,17,IOT(I+1))=IT4
     50.      C
     51.      C     SET LONGITUDE SIGN BIT.
     52.      C
     53.            FLD (17,1,IOT(I+1)) = 0
     54.            IF(IT5.GE.0) GOTO 3
```

```
      ITS=-ITS
      FLD(17,1,IOT(I+1))=1
C
C     PACK THE LONGITUDE.
C
    3 FLD(18,18,IOT(I+1))=ITS
C
C     IF I=3, WE ARE DONE.
C
      IF(I.EQ.3) RETURN
C
C     SET INTERMEDIATE VARIABLES TO FINAL POINT VALUES.
C
      IT1=FDAT
      IT2=FTIM
      IT3=FZON
      IT4=FLAT
      IT5=FLON
C
C
    4 CONTINUE
      END
```

```
@FOR,S@     N.PLANAR,.PLANAR

      SUBROUTINE PLANAR(VALUE,UNICDE,UNITS)
C
C     TABLE OF AREA MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,9),UNITS(9),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(9)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,9)/13,14,15,16/
C
C     PUT IN ALPHA UNITS.
      DATA ((ENTRY(I,J),I=1,2),J=1,9)/'CN**2 ','      ',
     .                                'METERS','**2   ',
     .                                'FEET**','2     ',
     .                                'YARDS**','2    '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='METERS'
      UNITS(4)=' SQ.  '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-14
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,9)/10000.000,1.000,7.524900,.836100/
C
C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,4
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
```

```
      C     ELSE, CHECK REST OF TABLE.
      C
    2 CONTINUE
      C
      C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
      C
            UNITS(1)='UNITS '
            UNITS(2)='ERROR '
            PRINT 101,ICODE
      101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN AREA TABLE.')
            RETURN
            END
```

```
@FOR,SW     N.POWOWR,,POWOWR

      1.          SUBROUTINE POWOWR(VALUE,UNICDE,UNITS)
      2.    C
      3.    C     TABLE OF POWER MEASURE CONVERSION FACTORS.
      4.    C
      5.          INTEGER ENTRY(3,2),UNITS(4),UNICDE
      6.    C
      7.    C     FACTORS ARE DOUBLE PRECISION.
      8.    C
      9.          DOUBLE PRECISION FACTOR(2)
     10.    C
     11.    C     PUT UNIT CODES INTO ENTRY.
     12.    C
     13.          DATA (ENTRY(I,I),I=1,2)/59,60/
     14.    C
     15.    C     PUT IN ALPHA UNITS.
     16.    C
     17.          DATA ((ENTRY(I,J),I=2,3),J=1,2)/'ERGS/S','EC,   ',
     18.         .              'WATTS ','      '/
     19.    C
     20.    C     ENTER INTERNAL UNITS.
     21.    C
     22.          UNITS(3)='WATTS/'
     23.          UNITS(4)='      '
     24.    C
     25.    C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
     26.    C
     27.          IF(UNICDE.EQ.0) UNICDE=-60
     28.    C
     29.    C     SET CONVERSION FACTORS.
     30.    C
     31.          DATA (FACTOR(I),I=1,2)/1.0D-7,100/
     32.    C
     33.    C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
     34.    C
     35.          UNITS(1)='UNITS '
     36.          UNITS(2)='ERROR '
     37.    C
     38.    C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
     39.    C          CODE INTO 'UNITS'.
     40.    C
     41.          ICODE=IABS(UNICDE)
     42.          DO 2 I=1,2
     43.          IF(ICODE.NE.ENTRY(1,I)) GO TO 2
     44.          IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
     45.          IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
     46.          UNITS(1)=ENTRY(2,I)
     47.          UNITS(2)=ENTRY(3,I)
     48.    C
     49.    C     AT THIS POINT, CONVERSION IS COMPLETE.
     50.    C
     51.          RETURN
     52.    C
     53.    C     ELSE, CHECK REST OF TABLE.
     54.    C
```

```
      2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN POWER TABLE.')
      RETURN
      END
```

```
@FOR,S:     N.PRESSN,.PRESSR

      1.          SUBROUTINE PRESSR(VALUE,UNICDE,UNITS)
      2.    C
      3.    C     TABLE OF PRESSR MEASURE CONVERSION FACTORS.
      4.    C
      5.          INTEGER ENTRY(3,9),UNITS(4),UNICDE
      6.    C
      7.    C     FACTORS ARE DOUBLE PRECISION.
      8.    C
      9.          DOUBLE PRECISION FACTOR(9)
     10.    C
     11.    C     PUT UNIT CODES INTO ENTRY.
     12.    C
     13.          DATA ((ENTRY(I,J),I=1,9)/48,49,50,51,52,53,54,55,56/
     14.    C
     15.    C     PUT IN ALPHA UNITS.
     16.    C
     17.          DATA (((ENTRY(I,J),I=2,3),J=1,9)/'MICRO ','BARS  ',
     18.         *                 'PASCAL','S     ','MICRO ','PAS   ',
     19.         *                 'KG/CM','**2  ','LBS/I','N.**2 ',
     20.         *                 'LBS./F','T.**2 ','IN. HE','RCURY ',
     21.         *                 'MM. HE','RCURY ','ATMOSP','HERES '/
     22.    C
     23.    C     ENTER INTERNAL UNITS.
     24.    C
     25.          UNITS(3)='KG/CM'
     26.          UNITS(4)='**2    '
     27.    C
     28.    C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
     29.    C
     30.          IF(UNICDE.EQ.0) UNICDE=-51
     31.    C
     32.    C     SET CONVERSION FACTORS.
     33.    C
     34.          DATA (FACTOR(I),I=1,9)/101.971D0,1.02D-3,1.02D0,1D0,.7031D0,
     35.         *                       4.882D-3,.3453627D0,8.7722125D0,
     36.         *                  1.033222887D0/
     37.    C
     38.    C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
     39.    C          CODE INTO 'UNITS'.
     40.    C
     41.          ICODE=IABS(UNICDE)
     42.          DO 2 I=1,9
     43.          IF(ICODE.NE.ENTRY(1,I)) GO TO 2
     44.          IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
     45.          IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
     46.          DO 1 J=1,2
     47.          UNITS(J)=ENTRY(J+1,I)
     48.        1 CONTINUE
     49.    C
     50.    C     AT THIS POINT, CONVERSION IS COMPLETE.
     51.    C
     52.          RETURN
     53.    C
     54.    C     ELSE, CHECK REST OF TABLE.
```

```
      C
    2 CONTINUE
      C
      C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
      C
          UNITS(1)='UNITS '
          UNITS(2)='ERROR '
          PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN PRESSURE TABLE.')
          RETURN
          END
```

```
@ELT,L     N.PRINTO,.PRINTJ

      DIMENSION D(28)
      INTEGER D
      CALL NTRAN(10,22)
      CALL SETADR(10,0)
      CALL NTRAN(10,2,20,D,LSTAT)
      CALL NTRAN(10,22)
C
      IF (LSTAT .GT. 0) GO TO 20
C
      WRITE (6,10) LSTAT
   10 FORMAT(' ERROR-- LSTAT =',I5)
      STOP
C
   20 IM = D(8) / 10000
      ID =(D(8) - IM * 10000) / 100
      IY =  D(8) - IM*10000 - ID*100
      FLD(18,18,IS) = FLD(0,18,D(10))
      FLD(18,18,IL) = FLD(18,18,D(10))
      FLD(18,18,IS1) = FLD(0,18,D(14))
      FLD(18,18,IL1) = FLD(18,18,D(14))
      FLD(18,18,IS2) = FLD(0,18,D(15))
      FLD(18,18,IL2) = FLD(18,18,D(15))
      FLD(18,18,IS3) = FLD(0,18,D(16))
      FLD(18,18,IL3) = FLD(18,18,D(16))
C
      WRITE(6,25) (D(I),I=1,7),IM,ID,IY,D(9),IS,IL,(D(J),J=11,13)
   25 FORMAT (/////,1X,7A6,/,' DATE OF LAST UPDATE = ',I2,'/',I2,'/',I2,
     *        /,' NEXT AVAILABLE SECTOR =',I10,/,' ADDRESS OF LAST EXPER
     *IMENT LEVEL REC. =',2I10,/,' LAST EXP. NO. =',I5,/,' NO. CRUISES R
     *EMAINING =',I5,/,' NO. STATIONS REMAINING =',I5)
      WRITE (6,26) IS,IL,IS1,IL1,IS2,IL2,IS3,IL3
   26 FORMAT (' ADDRESS OF LAST CRUISE LEVEL RECORD =',2I10,/,
     *        ' ADDRESS OF LAST ACOUSTIC STATION = ',2I10,/
     *,       ' ADDRESS OF LAST ENVIRONMENTAL STATION = ',2I10)
C
      STOP
      END
```

```
@FOR,SA     N,PRSLVL,,PRSLVL

      SUBROUTINE PRSLVL(VALUE,UNICDE,UNITS)
C
C     TABLE OF PRESSURE LEVEL MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(I,1),I=1,3)/57,70,71/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,3)/'DB//1U','BAR   ',
     *              '//.000','2D/CM2','DB//1U','PAS   '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='DB//1U'
      UNITS(4)='PAS   '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-71
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,3)/-100.000,-26.000,000/
C
C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,3
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE+FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE-FACTOR(I)
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
```

```
      2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN PRESSURE LEVEL TABLE.')
      RETURN
      END
```

```
@FOR,SN     N.SHRTME,.SHRTME

      1.          SUBROUTINE SHRTME(VALUE,UNICDE,UNITS)
      2.    C
      3.    C     TABLE OF SHRTME MEASURE CONVERSION FACTORS.
      4.    C
      5.          INTEGER ENTRY(3,5),UNITS(4),UNICDE
      6.    C
      7.    C     FACTORS ARE DOUBLE PRECISION.
      8.    C
      9.          DOUBLE PRECISION FACTOR(5)
     10.    C
     11.    C     PUT UNIT CODES INTO ENTRY.
     12.    C
     13.          DATA ((ENTRY(I,J),I=1,5)/21,22,23,24,25/
     14.    C
     15.    C     PUT IN ALPHA UNITS.
     16.    C
     17.          DATA (((ENTRY(I,J),I=2,3),J=1,5)/'MILLIS','ECONDS',
     18.         .                 'SECOND','S     ','MINUTE','S     ',
     19.         .                 'HOURS ','      ','DAYS  ','      '/
     20.    C
     21.    C     ENTER INTERNAL UNITS.
     22.    C
     23.          UNITS(3)='SEC.  '
     24.          UNITS(4)='      '
     25.    C
     26.    C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
     27.    C
     28.          IF(UNICDE.EQ.0) UNICDE=-22
     29.    C
     30.    C     SET CONVERSION FACTORS.
     31.    C
     32.          DATA (FACTOR(I),I=1,5)/.001D0,1.0D0,60.D0,3.6D3,8.64D4/
     33.    C
     34.    C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
     35.    C          CODE INTO 'UNITS'.
     36.    C
     37.          ICODE=IABS(UNICDE)
     38.          DO 2 I=1,5
     39.          IF(ICODE.NE.ENTRY(1,I)) GO TO 2
     40.          IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
     41.          IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
     42.          DO 1 J=1,2
     43.          UNITS(J)=ENTRY(J+1,I)
     44.        1 CONTINUE
     45.    C
     46.    C     AT THIS POINT, CONVERSION IS COMPLETE.
     47.    C
     48.          RETURN
     49.        2 CONTINUE
     50.    C
     51.    C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
     52.    C
     53.          UNITS(1)='UNITS '
     54.          UNITS(2)='ERROR '
```

```
      [10] FORMAT(' ERROR--UNIT CODE=',I3,' NOT IN SHORT TIME TABLE.')
      RETURN
      END
```

```
@FOR,S4     N.SPEEDS..SPEEDS

      SUBROUTINE SPEEDS(VALUE,UNICDE,UNITS)
C
C     TABLE OF SPEEDS MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,8),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(8)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,8)/28,29,30,31,32,33,34,99/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,8)/'CM/SEC','.    ',
     .                'METERS','/SEC. ','FT./SE','C.    ',
     .                'YARDS/','SEC.  ','KNOTS ','      ',
     .                'KM./HR','.    ','MILES/','HR.   ',
     .                'FRLNGS','/FRTNT'/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='METERS'
      UNITS(4)='/SEC. '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-29
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,8)/.01D0,1D0,.3048D0,.9144D0,.5145D0,.277777D0
     .                      ,.4469444D0,1.662D-9/
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C        CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,8
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
```

```
      2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN VELOCITY TABLE.')
      RETURN
      END
```

```
@FOR,S4     N.STOR,*STOR

      SUBROUTINE STOR (CODE,VALUE,UNIT,METHOD)
C
C         STOR (STORE) MOVES PARAMETERS INPUT ON THE ENTI CARD
C         TO THE CONSTANTS AREA OF /CPV/.
C
C
      IMPLICIT INTEGER (A-Z)
C
C
      COMMON /CPV    /NCON,CON(4,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
     *                NDATS,NNOBS(50),NDATA,VALRNG(2,30,50),DATA(30000)
      DIMENSION ICON(4,30),IPAR(30,30),IVLRNG(2,30,50)
      EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
     *            (VALRNG(1,1,1),IVLRNG(1,1,1))
      REAL CON,PAR,DATA,VALRNG
C
      REAL VALUE
      NCON = NCON +1
      ICON(1,NCON) = CODE
      ICON(2,NCON) = UNIT
      CON (3,NCON) = VALUE
      ICON(4,NCON) = METHOD
      RETURN
      END
```

```
@FOR,S  N.TMPTUR,.TMPTUR

      SUBROUTINE TMPTUR(VALUE,UNICODE,UNITS)
C
C     TABLE OF TMPTUR MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,9),UNITS(9),UNICODE
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,9)/35,36,97,98/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,9)/'FAHREN','HEIT  ',
     .                'DEG. C','ELSIUS','KELVIN','      ',
     .                'RANKIN','ES    '/
C
C     ENTER INTERNAL DEGREE UNITS.
C
      UNITS(3)='DEGREE'
      UNITS(4)='S CENT'
C
C     IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICODE.EQ.0) UNICODE=-36
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICODE)
      DO 1 I=1,4
      IF(ICODE.NE.ENTRY(1,I)) GO TO 1
      GO TO (2,3,4,5),I
    1 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
      RETURN
    2 IF(UNICODE.GE.0) VALUE=(VALUE-32.)*5./9.
      IF(UNICODE.LT.0) VALUE=VALUE*9./5.+32.
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
      RETURN
    3 UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
      RETURN
    4 IF(UNICODE.GE.0) VALUE=VALUE-273.15
      IF(UNICODE.LT.0) VALUE=VALUE+273.15
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
      RETURN
    5 IF(UNICODE.GE.0) VALUE=(VALUE-491.67)*5./9.
```

```
      IF(UNICODE.LT.0) VALUE=VALUE*9./5.+491.67
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
      RETURN
101   FORMAT(' ERROR--UNIT CODE ',I3,' NOT IN TEMPERATURE TABLE.')
      END
```

```
@ELT,L     N.UNPKCR,.UNPACR

      1.          SUBROUTINE UNPKCR(DATA)
      2.          IMPLICIT INTEGER(A-Z)
      3.    C
      4.          DIMENSION DATA(1),INOTES(1)
      5.    C
      6.          COMMON /CRUISE/CRUNO,PNXTC,CAILAT,CAILON,CAIDAT,CAFLAT,CAFLON,
      7.         *               CAFDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,
      8.         *               CASTAT(2),CESTAT(2),PACS,PENS,NCARDS,NOTES(14,132),
      9.         *               CAFLG
     10.    C
     11.          EQUIVALENCE(INOTES(1),NOTES(1,1))
     12.    C
     13.          CRUNO=FLD(0,18,DATA(2))
     14.    C
     15.    C     UNPACK NUMBER OF CARDS.
     16.    C
     17.          NCARDS=FLD(18,18,DATA(2))
     18.    C
     19.          PNXTC=DATA(3)
     20.    C
     21.    C     UNPACK ACOUSTIC INITIAL AND FINAL POINT BOUNDS.
     22.    C
     23.          CALL UPKREA(' ',CAILAT,CAILON,CAIDAT,0,' ',
     24.         *               CAFLAT,CAFLON,CAFDAT,0,' ',DATA(4))
     25.    C
     26.    C     UNPACK ENVIRONMENT INITIAL AND FINAL POINT BOUNDS.
     27.    C
     28.          CALL UPKREA(' ',CEILAT,CEILON,CEIDAT,0,' ',
     29.         *               CEFLAT,CEFLON,CEFDAT,0,' ',DATA(8))
     30.    C
     31.          CASTAT(1)=DATA(12)
     32.          CASTAT(2)=DATA(13)
     33.    C
     34.          CESTAT(1)=DATA(14)
     35.          CESTAT(2)=DATA(15)
     36.    C
     37.          PACS=DATA(16)
     38.          PENS=DATA(17)
     39.    C
     40.    C     UNPACK NOTES.
     41.    C
     42.          N=14*NCARDS
     43.          DO 1 I=1,N
     44.          INOTES(I)=0
     45.        1 INOTES(I)=DATA(I+17)
     46.          RETURN
     47.          END
```

```
@ELT,L      N.UNPKXP,.UNPKXP

      SUBROUTINE UNPKXP(DATA)
      IMPLICIT INTEGER(A-Z)
C
      DIMENSION DATA(1)
C
      COMMON /EXPMNT/EXNO,PNXTE,PCRU,EAILAT,EAILON,EAIDAT,EAFLAT,EAFLON,
     .               EAFDAT,EEILAT,EEILON,EEIDAT,EEFLAT,EEFLON,EEFDAT,
     .               EASTAT(2),EESTAT(2),EAFLG
C
C
C     UNPACK EXPERIMENT NUMBER.
C
      EXNO=DATA(2)
C
C     UNPACK ACOUSTIC INITIAL AND FINAL POINT BOUNDS.
C
      CALL UPKREA(' ',EAILAT,EAILON,EAIDAT,0,' ',
     .            EAFLAT,EAFLON,EAFDAT,0,' ',DATA(3))
C
C     UNPACK ENVIRONMENT INITIAL AND FINAL POINT BOUNDS.
C
      CALL UPKREA(' ',EEILAT,EEILON,EEIDAT,0,' ',
     .            EEFLAT,EEFLON,EEFDAT,0,' ',DATA(7))
C
      EASTAT(1)=DATA(11)
      EASTAT(2)=DATA(12)
C
      EESTAT(1)=DATA(13)
      EESTAT(2)=DATA(14)
C
      PNXTE=DATA(15)
C
      PCRU=DATA(16)
      RETURN
      END
```

```
@FOR,S@     N.VOLUME,.VOLUME

      SUBROUTINE VOLUME(VALUE,UNICDE,UNITS)
C
C     TABLE OF VOLUME MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,9),UNITS(9),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(9)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA ((ENTRY(I,J),I=1,3),J=1,9)/17,18,19,20/
C
C     PUT IN ALPHA UNITS.
C
      DATA (((ENTRY(I,J),I=2,3),J=1,9)/'CM  ','    ',
     .                 'METERS','  ','FEET  ','  ',
     .                 'YARDS ','  '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='METERS'
      UNITS(9)=' CU.  '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-18
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,9)/1.0D-6,100.,2.83170-2,.7646D0/
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,4
      IF((ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
    2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
```

```
      C
            UNITS(1)='UNITS '
            UNITS(2)='ERROR '
            PRINT 101,ICODE
        101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN VOLUME TABLE.')
            RETURN
            END
```

```
@ELT,L     N.#EST,.#EST

      1.          INTEGER FUNCTION #EST (A,B)
      2.    C
      3.    C
      4.          INTEGER A,B
      5.    C
      6.          IF (IABS(A-B) .GT. 180000) GO TO 10
      7.          #EST = MIN(A,B)
      8.          RETURN
      9.    C
     10.       10 #EST = B
     11.          IF (A .GT. 0) #EST = A
     12.          RETURN
     13.          END
```

APPENDIX B

CURRENT NAVDAB STEERING GROUP

APPENDIX B

CURRENT NAVDAB STEERING GROUP

The NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB) is under the general sponsorship and direction of NAVSEA 06H1-4. NAVDAB functions as a subsidiary of the Mobile Sonar Technology (MOST) Acoustics of the Medium Committee (ACME) for NAVSEA 06H1. Dissemination of information and requests for any of the documents pertaining to NAVDAB may be obtained by sending a formal request to any of the following current NAVDAB Steering Group members. A list of official mailing addresses follows the list of Steering Group members.

Executive Secretary

A. P. Franceschetti, SEA 06H1-4
Naval Sea Systems Command
Washington, D.C. 20362
Commercial (202) 692-3166
Autovon 222-3166

Chairman

L. A. Smothers, Code 3073
Naval Undersea Center
San Diego, California 92132
Commercial (714) 225-2316
Autovon 933-2316

Branch Managers

F. R. DiNapoli, Code TA113
Naval Underwater Systems Center
New London, Connecticut 06320
Commercial (203) 442-0771 Ext. 2647
Autovon 636-2647

Branch Managers (Continued)

W. H. Geddes, Code 3440
Naval Oceanographic Office
Washington, D.C. 20373
Commercial (202) 433-3994
Autovon 288-3994

L. A. Smothers, Code 3073
Naval Undersea Center
San Diego, California 92132
Commercial (714) 225-2316
Autovon 933-2316

Steering Group

F. R. DiNapoli, Code TA113
Naval Underwater Systems Center
New London, Connecticut 06320
Commercial (203) 442-0771 Ext. 2647
Autovon 636-2647

R. H. Ferris, Code 8120
Naval Research Laboratory
Washington, D.C. 20375
Commercial (202) 767-3359
Autovon 297-3359

W. H. Geddes, Code 3440
Naval Oceanographic Office
Washington, D.C. 20373
Commercial (202) 433-3994
Autovon 288-3994

R. F. Hosmer, Code 3071
Naval Undersea Center
San Diego, California 92132
Commercial (714) 225-2316
Autovon 933-2316

Steering Group (Continued)

W. A. Kuperman, Code 8120
Naval Research Laboratory
Washington, D.C. 20375
Commercial (202) 767-3210
Autovon 297-3210

K. V. Mackenzie, Code 9420
Naval Oceanographic Office
Naval Research Laboratory
4555 Overlook Ave. SW
Washington, D.C. 20375
Commercial (202) 767-2830
Autovon 297-2830

L. C. Maples, Code TA1A
Naval Underwater Systems Center
New London, Connecticut 06320
Commercial (203) 442-0771 Ext. 2501
Autovon 636-2501

S. R. Santaniello, Code TA111
Naval Underwater Systems Center
New London, Connecticut 06320
Commercial (203) 442-0771 Ext. 2675
Autovon 636-2675

J. A. Whitney, Code 3073
Naval Undersea Center
San Diego, California 92132
Commercial (714) 225-2316
Autovon 933-2316

OFFICIAL MAILING ADDRESSES

1. Commander
   Naval Undersea Center
   San Diego, California 92132

2. Commanding Officer
   Naval Underwater Systems Center
   New London, Connecticut 06320

3. Director
   Naval Research Laboratory
   Washington, District of Columbia 20375

4. Commander
   Naval Oceanographic Office
   Washington, District of Columbia 20373

5. Commander
   Naval Sea Systems Command
   Washington, District of Columbia 20362

USER COMMENTS FORM

Comments concerning this document are invited. Please send one of the following attached forms or a letter to:

L. A. Smothers, Code 307
Naval Undersea Center
San Diego, California 92132

(Envelope address: Commander, Naval Undersea Center, San Diego, California 92132)

MEMORANDUM Date: _____

From: _____

_____

_____

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),
SEA 06H1/036-EVA/MOST- 4 (Volume 3: Details of Creation Phase)

CUT

MEMORANDUM Date: _____

From: _____

_____

_____

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),
SEA 06H1/036-EVA/MOST- 4 (Volume 3: Details of Creation Phase)

CUT

MEMORANDUM

From: _____ Date: _____

_____

_____

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),
SEA 06H1/036-EVA/MOST-4 (Volume 3: Details of Creation Phase)

CUT

MEMORANDUM Date: _____

From: _____

_____

_____

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),
SEA 06H1/036-EVA/MOST-4 (Volume 3: Details of Creation Phase)

DISTRIBUTION LIST

5 Naval Sea Systems Command
NSEA-06H1 (5 cys)

3 Naval Oceanographic Office
Code 3432
Code 3440
Code 6100

2 Naval Research Laboratory
Code 8120 (2 cys)

7 Naval Undersea Center, San Diego
Code 307 (7 cys)

6 Naval Underwater Systems Center, New London Laboratory
Code TA1A
Code TA42
Code TA111 (2 cys)
Code TA112
Code TA113

2 Defense Documentation Center